PCT
WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 5 : G06F 15/62

A1

(11) International Publication Number: WO 93/12501

(43) International Publication Date: 24 June 1993 (24.06.93)

(21) International Application Number: PCT/US92/10560

(22) International Filing Date: 8 December 1992 (08.12.92)

(30) Priority data:
07/805,448 10 December 1991 (10.12.91) US

(71) Applicant: LOGITECH, INC. [US/US]; 6505 Kaiser Drive, Fremont, CA 94555 (US).

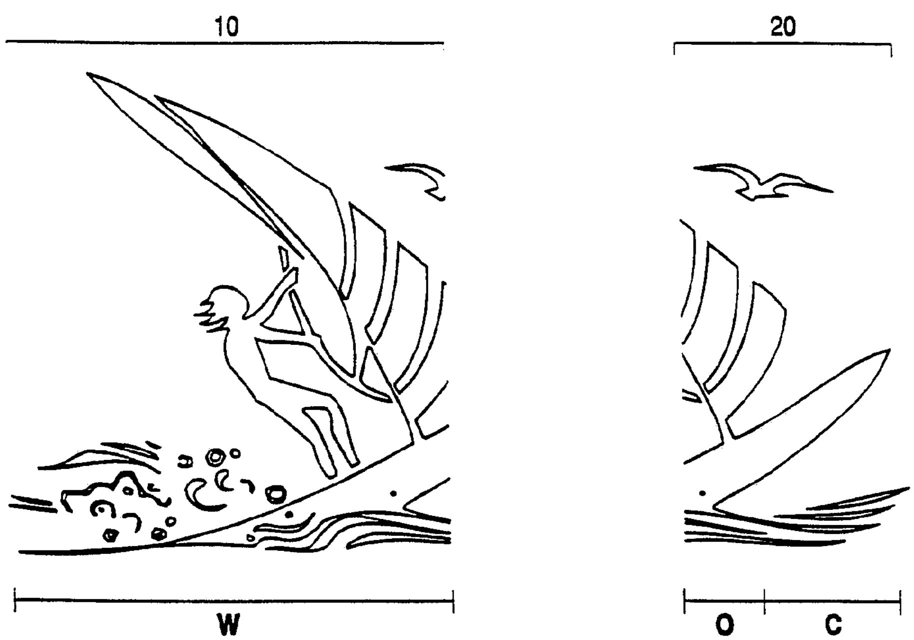
(72) Inventors: PITTELOUD, Dominique ; 917 Sycamore Street, Palo Alto, CA 94303 (US). PHILLIPS, Peter ; 817 Lincoln Avenue, Palo Alto, 94301 (US). MARTIN, Jean-Paul, H. ; 10368 Granite Creek Lane, Oakton, VA 22124 (US).

(74) Agent: HAUGHEY, Paul, C.; Townsend and Townsend, One Market Plaza, 20th Floor, Steuart Tower, San Francisco, CA 94105 (US).

(81) Designated States: AU, CA, JP, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: APPARATUS AND METHODS FOR AUTOMERGING IMAGES



(57) Abstract

According to the invention, apparatus and methods are provided for automatically merging a variety of images, including vector and bitmap graphic images, as well as text images. In a method of the present invention, the automatic merging of image strips (10, 20) to recreate a source image includes acquiring first (10) and second (20) image strips; locating at least one feature in the first (10) image strip; locating this feature in the second (20) image strip; and merging the two image strips (10, 20) into a single image, based on the matching feature.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria
AU	Australia
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CS	Czechoslovakia
CZ	Czech Republic
DE	Germany
DK	Denmark
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IT	Italy
JP	Japan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakhstan
LI	Liechtenstein
LK	Sri Lanka
LU	Luxembourg
MC	Monaco
MG	Madagascar
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SK	Slovak Republic
SN	Senegal
SU	Soviet Union
TD	Chad
TG	Togo
UA	Ukraine
US	United States of America
VN	Viet Nam

APPARATUS AND METHODS FOR AUTOMERGING IMAGES

BACKGROUND OF THE INVENTION

The present invention relates generally to the field of computer graphics and, more particularly, to methods and apparatus for merging graphic images.

Computers have become a powerful tool for the rapid and economic creation of pictures, and are particularly well suited for drafting and publishing tasks. For drafting, computer-aided design or "CAD" has found wide application. Instead of pen and paper, the CAD user employs a computer with a pointing device (typically a mouse, light pen, digitizer tablet, or the like) to create an electronic description of a drawing in the computer's memory.

For publishing, desktop publishing or "DTP" has become the standard. In DTP, both text and graphic images are manipulated on a computer screen, much as they are in CAD. In either system, when the user is satisfied with the drawing, he or she obtains a "hard copy" by sending the stored image to an output device, such as a laser printer or plotter.

In working with graphic images, it is often desirable to "acquire" images and represent them in a computer. For example, a DTP user may wish to import a photograph into a newsletter. Before an image is available to the user, however, it must be converted to a format which the computer may interpret.

Thus, the process of entering images into a computer requires that an electronic description be generated. There are two basic approaches to describing images: vector objects and bitmaps. Vector objects are mathematical descriptions of an image. For example, a line may be described by its starting and ending points. A circle may be described by its center and radius. Of particular interest to the present invention, however, are bitmap images.

In bitmaps, an image is described as a two dimensional array of bits or pixels (picture elements). By arranging combinations of black and white pixels (0 and 1 bits), a monochromatic image may be reproduced. This technique is commonly employed to reproduce images for newspapers and magazines. Bitmaps are almost always rectangular, and if the image itself is not rectangular, then the image must be rendered with a "mask" bitmap that defines the shape of the image.

While the bulk of a bitmap comprises the bits themselves, header information is also necessary to define how the bits are interpreted. This includes the height and width of the bitmap (expressed in number of pixels) and color information. In a monochromatic bitmap, one bit in the bitmap corresponds to one pixel on the display. A color bitmap requires multiple bits per pixel. In this case, the header information describes how the multiple bits correspond to particular colors. There are multiple file formats for storing bitmaps. Examples include ZSoft's PC Paintbrush (PCX), CompuServe's Graphics Interchange Format (GIF), and Microsoft's Tagged-Image File Format (TIFF).

Because bitmaps are used to store real-world images, they usually enter a computer through a scanner or a video frame grabber. A scanner converts a photographic image into a bitmapped data file; similarly, a frame grabber converts a video signal (from a video camera or VCR) to a bitmapped data file. Bitmaps can also be created by hand using computer "paint" programs, such as Microsoft's Windows™ Paintbrush program. Once converted into a bitmap, an image may be

transferred to other computers in the same manner as other binary files (e.g., via magnetic media or modem).

Scanners, which are probably by far the most popular means for importing images, employ a light source and photodetectors to "read" or "scan" images into a computer. Two basic types are available: flatbed and handheld. The general construction and operation of each will now be briefly described.

Flatbed scanners look and operate very much like an ordinary photocopier. First, the user places an image to be scanned upon the flatbed (flat glass). Next, the scanner is activated, e.g., by pressing a button. The image is then scanned by a light source. Instead of generating a photocopy, however, the scanner focuses the image onto photodetectors which produce binary data representative of the image. Upon completion of the scan, the data are stored on a computer disk as a binary file, typically in one of the aforementioned bitmap file formats. The data file is now available to the user for use in application software, such as desktop publishing packages, paint programs, and the like.

In "handheld" scanners, on the other hand, substantially all of the scanning system is housed within a single handheld unit. Instead of placing an image to be scanned on a glass surface, as one would do with a photocopier or flatbed scanner, the image is laid face up on a flat surface, typically a desktop. The scanner, which includes a roller or wheels, is dragged across the surface of the image. The image is then recorded in a manner similar to that for flatbed systems.

While scanners allow for the easy importation of a variety of images, they have a significant limitation. Scanners, particularly handheld models, have a limited scanning area or "window" for accommodated materials to be scanned. If the image to be scanned is larger than the viewing window of the scanner, then the image cannot be captured in a single operation. In this instance, the desired image must be scanned as a series of multiple strips, i.e., portions which are small enough to fit within the viewport of the scanner. After

scanning all the strips, the image may be reconstructed within the computer by manually "stitching" the strips back together (as described hereinbelow).

Before a user may stitch strips back together, other problems must be overcome. In particular, the hand scanner's reliance upon the human hand as a means for moving the unit across a source image creates artifacts. For example, strips obtained from the same image will typically have different lengths. Moreover, the geometric orientation between strips often differ in horizontal alignment (X translation), vertical alignment (Y translation), rotation, and the like. In addition, one or more strips may undergo varying rates of uniform compression and/or expansion, depending on how fast or slow the user has moved the scanner across the image. To faithfully reproduce the source image, therefore, these artifacts must be eliminated or minimized.

Even if the difficulty of artifacts is overcome, the process of stitching strips together to form the original image is cumbersome. For example, a common technique for stitching strips together requires the user to "eyeball" (i.e., judge with one's eyes) the alignment between the bitmap strips. The technique is far from perfect, however, as this requires an elaborate trial and error technique to obtain an acceptable image. In addition, the current process does not correct for the difference in documents or images because of "stretching" (transformation) due to a fast scanning speed. Moreover, the user must often manually correct for vertical skewing (rotation) of each image strip. Routinely, the user obtains a bitmap where the intersections of the strips (i.e., the stitched areas) appear rough and misaligned.

Another technique requires the use of registration marks, i.e., marks that are added to the source image that allow the software to find the marks and merge the image. The marks are generally added to an image by scanning the image with a transparency having the marks overlaid on the image. The software then finds the marks and attempts to merge the images based upon the location of the marks. Another technique employs a plastic grid to aid the user in scanning images in

strips in a more consistent manner. However, none of these techniques are transparent to the user, nor are they fully automated. Moreover, these techniques rely heavily on the dexterity of the individual user in aligning images or adding registration marks to the image that the software can find and use to merge the images. The results are almost always less than satisfactory.

Thus, it is desirable to provide a system and methods which automatically align bitmap images with minimum user effort. Moreover, the methods employed should achieve precise alignment between two or more images -- precision that cannot be achieved by manual positioning techniques alone. The present invention fulfills this and other needs.

SUMMARY OF THE INVENTION

Computers have found wide application for the creation and editing of drawings or graphic images. A particularly easy technique for creating images is to scan a drawing into a computer using one of the commercially available scanners. Because scanning devices, particularly handheld scanners, have a limited viewport or viewing window for obtaining images, prior art systems had required the computer user to expend significant effort and time manually piecing or stitching scanned images together for those images which do not fit within the viewport.

According to the present invention, a method for automatically merging images on a computer system includes acquiring first and second image strips; locating at least one feature in the first image strip; locating this feature in the second image strip; and merging the two image strips into a single image, based on the matched features. Additional strips may be acquired and merged as desired. For better results, a plurality of features are matched between the image strips.

A system for automerging images, constructed in accordance with the principles of the present invention, includes a computer having a memory and a processor; a scanner for acquiring an image as a series of image strips; means for locating at least one feature in the first image strip; means

for locating this feature in the second image strip; and means for merging the two image strips into a single image by aligning the two features.

BRIEF DESCRIPTION OF THE DRAWINGS

Figs. 1A-C illustrate the process of acquiring and merging bitmaps using manual techniques.

Figs. 1D-F illustrate artifacts which are common in merged bitmaps.

Fig. 2 is a block diagram of a computer system in which the present invention may be embodied.

Figs. 3A-B illustrate a window interface or work surface of the present invention for manipulating images.

Fig. 3C illustrates a scan mode panel of the present invention.

Fig. 4 is a flow chart of the Automerge method of the present invention.

Figs. 5A-B are a flow chart of the M_MergeFind method of the present invention.

Fig. 5C illustrates the process of sampling pixel blocks from a select region.

Fig. 6A is a flow chart of the Autocorrelate method of the present invention.

Fig. 6B illustrates the movement of pixels during the Autocorrelate method.

Fig. 7 is a flow chart of the M_MergeMatch method of the present invention.

Figs. 8A-B are a flow chart of the Coarse/Fine Correlation of the present invention.

Figs. 9A-B are a flow chart of the Rank_Features method of the present invention.

Fig. 9C illustrates the pairing of features and matches within each image strip.

Fig. 9D illustrates the calculation of stretching between image strips.

Fig. 9E illustrates the calculation of multiple regions of compression and/or expansion between image strips.

Fig. 10A is a flow chart of the Calculate Dimensions method of the present invention.

Figs. 10B-C illustrate the calculation of rectangles and matrix operations for merging.

Fig. 11 is a flow chart of the M_MergeImage method of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Introduction

Referring now to Figs. 1A-F, a conventional technique for merging a plurality of image strips will be described. If the image strips are not already digitized as a bitmap, then the strips must first be acquired. As shown in Fig. 1A, for example, a source bitmap image 5 may be acquired by scanning with a handheld scanner 105. Since image 5 is wider than the maximum width **W** of scanner 105, however, it cannot be acquired in a single pass. In particular, an area **C**, which beyond the viewing window of the scanner 105, will be cropped. Thus, the image must be scanned as two separate strips. Typically, the first strip will occupy the entire width **W** of scanner 105; the second strip will include the area cropped **C** as well as a small overlap region **O**.

As shown in Fig. 1B, two image strips 10, 20 are obtained (in separate operations) from the single image 5. Image strip 10 has a width of **W** (the maximum width of the scanning window), with the portion of the image 10 which lies outside of this effective area being cropped. Thus, the user must obtain an additional strip, such as the strip 20, which includes the portion of the source image which was cropped **C** as well as an overlapping region **O**. For simplification, image strips 10, 20 are shown without skewing, compression, and/or expansion artifacts, which typically will be present in images obtained with a hand scanner.

With particular reference to Figs. 1C-F, the merging of image strips 10, 20 will be described. As shown in Fig. 1C, one image strip (e.g., strip 20) is moved towards the other. Typically, the user first selects a stitching (or similar) mode. Next, a special cursor, such as cursor 22, is displayed

to assist the user in the operation. In response to user-generated signals, one image strip (e.g., strip 20) is moved or "dragged" in a direction 15 towards the other image strip (e.g., strip 10); the signals may be supplied by a pointing device (e.g., a mouse) moving the cursor 22 in a desired direction. In this manner, signals from the pointing device are used to move or translate the image strip 20 towards the image strip 10.

Referring now to Fig. 1D, the completed image 30 is shown. Image 30 is a single image, such as a bitmap, which results from the merging of strips 10, 20 (at intersection I). While the image is a fair reproduction of the original, it is misaligned at the intersection I of the two image strips. As shown in Figs. 1E-F, the misalignment within the image can best be seen from enlarged sections 31, 35 (taken from Fig. 1D). In section 31, for example, the intersection I of the strips is misaligned at point 32. Additional alignment artifacts are shown in section 35 where misalignment occurs at points 36, 37, 38.

For purposes of illustration, the foregoing example has been confined to simple translational artifacts -- in this case, horizontal (X) translation of one strip relative to another. In addition to translation artifacts, however, the strips used to reconstruct a single image will often have undergone a complex combination of other transformations, including skewing, compression, and/or expansion artifacts. Hence, merging images by conventional techniques routinely yields unsatisfactory results.

Preferred Embodiments

1. Acquisition of images

The invention may be embodied on a computer system such as the system 100 of Fig. 2, which comprises a central processor 101, a main memory 102, an I/O controller 103, a screen or display 104, a scanner 105, a mass storage device 106, a keyboard 107, a pointing device 108, and an output device 109. The various components of the system 100 communicate through a system bus 110 or similar architecture.

In operation, the user enters commands through keyboard 107 and/or pointing device 108, which may be a mouse, a track ball, a digitizing tablet, or the like. The computer displays text, graphic images, and other data through screen 104, such as a cathode ray tube. A hard copy of the image may be obtained from output device 109, which is typically a printer or a plotter. In a preferred embodiment, an appropriately programmed IBM PC-compatible personal computer (available from International Business Machines, Corp. of Armonk, NY) is used running under MS-DOS and Microsoft Windows™ (both available from Microsoft, Corp. of Redmond, WA).

In this interactive computer system, the user acquires graphic images from a variety of sources. Common sources includes input devices (e.g., scanner), software (e.g., paint programs), disk files (e.g., TIFF, PCX, and GIF formats), and the like. In typical operation, images are acquired with the scanner 105, which may be either a flatbed or handheld scanner. Scanners suitable for use as the scanner 105 are available from a variety of vendors; in a preferred embodiment, scanner 105 is a ScanMan™ 256, available from Logitech Corp. of Fremont, CA. Once entered into the computer 100, an image is stored as a bitmapped graphic and, thus, may be represented in the computer's memory 102.

Referring now to Fig. 3A, the system 100 provides a window or workspace 120 for display on screen 104. Window 120 is a rectangular, graphical user interface, running in Microsoft Windows™, for viewing and manipulating graphic images. Window 120 contains a plurality of menus 140, 150, 160, each having submenus and software tools for use on graphic images. Of particular interest to the present invention is an Acquire tool 151, which is available from menu 150. Window 120 also includes a client area 130 for displaying images, such as the bitmapped graphic image 131.

Referring to Fig 3B, the operation of window 120 for acquiring a graphic image will now be illustrated. The user initiates the image acquisition, for example, by selecting the Acquire option 151 from the menu 150 of Fig. 3A. In response,

the system 100 displays a scanning window 220. Window 220 includes interface components or resources, such as a scanning preview window 230, a ruler bar 250, a scan mode panel 240, radio buttons 260, dialogue buttons 280, and a status line 270.

Next, the user selects a "scan mode" from panel 240 which indicates the type (i.e., single or multiple strips) and orientation (i.e., portrait or landscape) of the image to be scanned. As shown in detail in Fig. 3C, panel 240 includes a plurality of icons arranged into two rows: single-strip scan 241 and multiple-strip scan 245. From single-strip scan 241, the user may select from a variety of single-strip modes, including 1) single-strip portrait 242 (top-to-bottom), 2) single-strip landscape 243 (left-to-right), and 3) single-strip landscape 244 (right-to-left). Similarly, from multiple-strip scan 245, the user may select from 1) multiple-strip portrait 246 (top-to-bottom), 2) multiple-strip landscape 247 (left-to-right), and 3) multiple-strip landscape 248 (right-to-left).

A particular advantage of the user interface provided by the scan mode panel 240 is efficiency of user input. Specifically, a variety of different images (e.g., single or multiple, portrait or landscape, and the like) may be acquired as a single operation. For example, to acquire an image as two vertical strips the user need only select multiple-strip portrait 246, scan the first strip, select stop (not shown), scan the second strip, and select Done (from buttons 280). Additional features and advantages of the interface are set forth hereinbelow as Appendix B.

Other scanning parameters may also be set at this point. The viewing width W of the scanner may be shortened, for example, by selecting positions on the ruler bar 250. Additional scanning parameters (e.g., grey scale or line art) may be entered by activating the "Options" button of buttons 280. If no parameters are specified by the user, however, system 100 will assume default values (e.g., single scan of line art).

Next, the user actually scans the source image by activating the scanner 105, typically by pressing a button or switch device located on the scanner. Upon activation, the

scanner 105 is ready to capture or acquire the source image; thus at this point, the user drags the scanner 105 across the source image in a smooth and continuous motion. Immediate user feedback is provided by preview window 230 which displays the acquired image in real-time.

On completion of the scan, the user selects "Done" from the buttons 280, typically by pressing the "D" key or selecting the button with the pointing device. At this point, the acquired image is stored in memory 102. If the user is not satisfied with the acquired image shown in window 230, the user may select "Rescan" from the buttons 280 and repeat the scanning process. Otherwise, the image will typically be saved to non-volatile storage 106 as a bitmap (e.g., TIFF) file. After two or more image strips have been acquired, the source image may be reconstructed by automerging techniques of the present invention.

2. Automerge: Automatic merging of images

The following description will focus on the automerging of two bitmap image strips obtained by a handheld scanning device. However, the present invention is not limited to such image formats or devices. Instead, a plurality of image strips having any one of a number of image formats (including both bitmap and vector formats) may be automatically merged in accordance with the present invention. Additionally, the image strips may be entered into the system 100 in a variety of formats (e.g., vector formats) and by a variety of means (e.g., file transfers, paint programs, and the like).

a. General operation

The automerging of images in accordance with the present invention will now be described. In an exemplary embodiment, system 100 operates under the control of an Automerge routine to combine two or more image strips together to recreate the source image. The routine, which is typically loaded into memory 102 from storage 106, instructs or directs processor 101 in the automatic merging of the image strips.

In a preferred embodiment, the Automerge routine is implemented in a message-passing environment (e.g., Microsoft

Windows™); thus, the routine is invoked in response to events received from an event handler. The dispatching of messages in an event-based system, such as Microsoft Windows™, is known in the arts; see, e.g., Petzold, C., *Programming Windows*, second edition, Microsoft Press, 1990, the disclosure of which is hereby incorporated by reference. Upon invocation, the Automerge routine will invoke or call (directly or indirectly) additional routines, including M_MergeFind, Autocorrelate, M_MergeMatch, Coarse/Fine Correlation, Rank_Features, Calculate Dimensions, and M_MergeImage routines.

Referring now to Fig. 4, the general operation of the Automerge routine 400 is illustrated by a flow chart. In step 401, the image strips (usually two at a time) are obtained. Typically, the image strips are obtained as bitmaps by hand scanning a source image (as previously described).

In step 402, system 100 is initialized for merging images. In particular, data structures are initialized in memory 102 for processing the images. For purposes of illustration, each image may be defined by the following record (struct), written in the C programming language:

```
typedef struct {
   short Width, Height;
   IMAGECLASS Class;
   FPGETLINE GetLine;
   short Handle;
} IMAGEINFO;
```

where Width and Height specify the dimensions of the image; Class, an enumerated variable of type IMAGECLASS, specifies whether the image is bilevel, grayscale, or color; GetLine is a pointer to a function which returns (via a pointer to a buffer) the pixel data for a single line in the image; and Handle serves as a handle or index of the image. Additional exemplary data structures for processing image strips are set forth hereinbelow in Appendix C.

In step 403, a "feature extraction" is performed for the first strip by the M_MergeFind routine (described hereinbelow in Fig. 5), which extracts two or more distinct features of interest from an area of the first strip which overlaps with the second strip (e.g., overlap O from Fig. 1). The method finds distinct areas or "features," i.e., those

areas which have the most amount (highest score) of "uncorrelation" in a given or surrounding neighborhood. To improve automerging, groups of features are sought in different areas of the first strip. Ideally, the features should be unique within the search area of the strip. The technique may be customized for a particular class of images (e.g., gray scale or bilevel line art). For example, the actual number of features, the area and number in the strip for the features to be searched, and the size of a feature are all parameters that can be customized to the image type.

In step 403, features found in the extraction step (i.e., from the overlapping area of the first strip) are located or detected in the second strip by the M_MergeMatch routine (described hereinbelow in Fig. 7). For detection, the routine employs a sequential similarity detection algorithm (SSDA) -- an optimized variant of normalized correlation or pattern matching -- which is known in the art. For each group of features found in the first strip, M_MergeMatch seeks a matching group in the second strip. Each feature/match pair is given a score describing how well it matches, how unique the match is, and how far apart a match has occurred (between first set and second set of pairs). The pairs with the best scores will be used in step 404 to perform the actual merging of image strips.

In step 404, the geometry of the second strip is normalized to the geometry of the first strip. Specifically, using the best features extracted from the first strip and the corresponding best matches in the second strip, the second strip is transformed into the geometry of the first strip. Exemplary transformations of the second strip include rotation, compression or expansion, shearing, and the like.

Finally, in step 405, the two image strips are merged together by the M_MergeImage routine. The basic process includes mapping the second strip (now normalized) into the first strip. The actual mapping is accomplished by matrix transformations (set forth in further detail hereinbelow). At the conclusion of step 405, the merged image is displayed in client area 130 (of Fig. 3A).

b. Specific operation

Referring now to Figs. 5-11, the individual components of the Automerge method will now be described in further detail. In Figs. 5A-B, the M_MergeFind method or routine 500 is represented by a flow chart. The routine is invoked with an image strip, a list of regions, and a number of features to find in each region **N**. The method returns a list of features (as Feature_List).

The individual steps of the method are as follows. In step 501, a loop is established to examine each region in the list for features. As shown in Fig. 5C, a region **R** is an area, typically a rectangle, defined within the overlap area. If an additional region to be examined exists at step 501, then at step 503 a threshold, a sorting list, and a feature list are initialized. The threshold is a minimum score for a feature. If an area under examination has a score below the threshold, the area will not be retained for further examination. While the threshold is initially set to a preselected value, the threshold will typically be adjusted upward, during runtime, to be no less than the score of features already found. The sorting list is a local data structure, typically implemented as a linked list. As features are located, they are stored, according to rank, in the sorting list. In a preferred embodiment, the sorting list has a finite limit of, for example, four members. Thus, features having a low score are automatically excluded. After examining a given region, the sorting list will contain the best (i.e., highest scoring) features located. By appending the sorting list to the feature list, the best features within each region are identified. Once all regions have been examined (no at step 501), then the method concludes by returning the feature-list at step 502. Otherwise, the method continues on to step 504.

At step 504, the search for features in a region begins by setting the block coordinates to the top left position of the region. In step 505, a block of data is retrieved from the first strip based on the width and height of the feature (Feature_Width and Feature_Height). This determines the bounds of a region, which will typically define

a square sample block **S** (of Fig. 5C). In a preferred embodiment, the sample block **S** is a 16X16 array of pixels. However, any convenient array size may be employed. In addition, the region may be divided into smaller rectangular regions (the size of a feature) to determine if it is a good feature (i.e., a good match).

In step 506, the Autocorrelate routine (described in further detail hereinbelow) is invoked to determine a score or Autocorrelate index for a given sample block. The method determines the "uniqueness" of the feature in a sample block by a series of image move and difference operations. If in step 507, the Autocorrelate index is greater than the threshold, the method stores the block position and index in the sorting list at step 510. If the sorting list is full at step 511, then the method removes the block with a smaller index at step 512 and resets the threshold to the minimum index in the list at step 513. After step 513, the method continues on to step 508 to move the block coordinates to the right; the method also proceeds directly to step 508 when the autocorrelate index is less than the threshold (i.e., no at step 507). At step 508, the method moves the block coordinates to the right, i.e., gets the next sample block.

In step 509, if the right limit (i.e., the right edge) of the region has been reached, then the block coordinates for the image strip are moved downward at step 514. Otherwise (no at step 509), the method loops to step 505 and continues as before. At step 515, the image is tested to determine whether the bottom limit (i.e., bottom edge) of the region has been reached. If the bottom limit of the region has not been reached (no at step 515), then the method loops back to step 505 and continues. However, if the edge has been reached (yes at step 515), then at step 516 the sorting list is appended to the feature list and the method loops back to step 501 to begin examination of the next region.

Referring now to Figs. 6A-B, the Autocorrelate method, which is called in step 506 above, is illustrated. In the Autocorrelate method 600, a sample block of the image is examined to determine if it is a feature. This is determined

by taking correlations of a sample block with a version of the block which has been moved. Each individual correlation must be larger than the current correlation threshold.

The individual steps of the method 600 will now be described in turn. In step 601, the current image block under examination is copied to a temporary data structure, such as a temporary array. In step 602, the data in the block array is shifted or moved downward, as illustrated by block 651 of Fig. 6B. In step 603, an index or score of correlation is determined from the absolute value of the subtraction of the shifted version from the original. The operation may be summarized by the following equation:

$$\%Index = \sum_{y}\sum_{x} |\, block(x,y) - block(x,y+1) | \tag{1}$$

where **x** and **y** represent the horizontal and vertical coordinates, respectively, for a given point in the block.

In step 604, if the index is not less than the threshold, then in step 605 the data in the array is shifted to the right, as shown by block 652 of Fig. 6B. However, if the index is less than the threshold (yes at step 604), then the method concludes. After step 605, the index is calculated in step 606 by again determining the difference between the shifted version of an image block from the original block:

$$\begin{aligned}&\%Index = \\ &MIN(\%Index, \sum_{y}\sum_{x} |\, block(x,y) - block(x+1,y+1) |)\end{aligned} \tag{2}$$

In step 607, again the index is tested to determine whether it is less than the threshold; if it is not, then in step 608 the data and the array is shifted upward, as shown by block 653 of Fig 6B. Otherwise (yes at step 607), the method concludes. After step 608, the index is again calculated in step 609 as follows:

$$\%Index = MIN(\%Index, \sum_{y}\sum_{x}|block(x,y) - block(x+1,y)|) \quad (3)$$

In step 610, the block is again shifted upward (block 654 of Fig. 6B) and the index calculated. In essence, step 610 repeats steps 607-609. Additional correlation operations (e.g., block 655), may be performed, as desired. In step 611, the index is returned.

Referring now to Fig. 7, the M_MergeMatch method or routine 700 is illustrated. In this method, for each feature found in the right strip, a corresponding feature is sought in the left strip. A list of matches in the second strip is then returned. Each step will now be described in further detail.

In step 701, a loop is established to examine all features in the Feature_List (i.e., the features returned from the first strip). If an additional feature exists, then in step 702 an estimate is made as to the position of a match in the second strip (based on the position of the feature in the first strip). In step 703, the size and position of the region to search is calculated based on the estimated position of the match.

In step 704, a bitmap of the feature of interest (from the first strip) is obtained. In step 705, the feature is filtered (e.g., by a low-pass filter) to smooth out its features. In step 706, a coarse correlation is made by invoking the Coarse Correlation method (set forth hereinbelow), which returns the best matches of the region using a subsample version of the feature under examination. Unlike a Fine Correlation method (described hereinbelow), the Coarse Correlation method only examines a select number of pixels (e.g., every other pixel).

In step 707, another loop is established to examine all coarse matches found (by the previous step). If another coarse match exists at step 707, then in step 708 a small region is defined around the match, and in step 709 the true position of the match is determined by calling the Fine Correlation method.

After fine correlation, the method loops back to step 707 for additional coarse matches. However, if another coarse match does not exist at step 707, then the method continues to step 710 where the best match (and associated features) is stored in a Match_List. The method then loops back to step 701 to process another feature. If an additional feature does not exist at step 701, however, then the method continues on to 711 where the features are ranked by the Rank_Features method (described hereinbelow). In step 712, the dimensions of the M_MergeImage are calculated and the method then concludes.

Referring now to Figs. 8A-B, the Coarse/Fine correlation method 800 is illustrated by a flow chart. In the method 800, for a given feature and a given region of the second strip, a list of N best matches is returned, with the correlation performed at either coarse or fine resolution. As such, the operation of the method is very similar to that of the Auto-Correlation method. Instead of correlating an image block with a version of itself, however, method 800 seeks to correlate a feature from the first strip with an image block in the second strip. The steps of the method will now be described in detail.

In step 801, threshold and Sorting List data structures (previously described) are locally initialized. At step 802, the block coordinates for the strip are set to the top left position of the region. At step 803, the size of the second strip is determined. In step 804, if a coarse correlation is desired, then in step 805 a subsample block and feature bitmap are obtained. Otherwise (no at step 804), step 805 is skipped. At step 806, index is calculated from the equation:

$$\%Index = \sum_{y}\sum_{x} |\,block(x,y) - Feature(x,y)\,| \qquad (4)$$

At step 807, if the index is less than threshold, then in step 808 the block is moved to the right. Otherwise (no at step 807), the procedure jumps to step 813 (shown in Fig. 8B).

In step 813, the block position and index are stored in the Sorting_List. In step 814, if the list is full, then the method continues on to steps 815 and 816. If the list is not full however, then the procedure continues on to step 808. In step 815 the block with the biggest index is removed from the list. In step 816 the threshold is calculated from the maximum indices in the list. After step 816, the method jumps back to step 808.

In step 808, the block is moved to the right. At step 809, the block is tested to determine whether it is at the right limit of the region. If it is, then the block is moved to the right at step 810; otherwise (no), the method returns back to step 802. At step 811, if the right limit of the region has been reached, then the method returns the sorting list at step 812 and then concludes. If the right limit has not been reached (no at step 811), then the method loops back to step 802.

Referring now to Figs. 9A-B, the Rank_Features method 900 is illustrated. In general, for a given list of features and a list of corresponding matches, the method returns the best pair of feature/match combination.

The steps of the method will now be described in further detail. In step 901, the features are grouped in pairs. As shown in Fig. 9C, for example, possible pairs for the left strip include L1/L2, L1/L3, L2/L3, and so forth. In step 902, a loop is established for all pairs. If an additional pair exists in 902, then in step 903 the distance between members of the pair is determined. For example, for feature pair L1/L2 and match pair R1/R2, the distance may be determined:

$$\begin{aligned} \%F_{Dist} &= \left| L1 - L2 \right| \\ \%M_{Dist} &= \left| R1 - R2 \right| \end{aligned} \tag{5}$$

In step 904, if either distance is less than a preselected minimum, then the method loops back to step 902. Otherwise, the method continues on to step 905 to determine theta -- the angle between a line joining feature pairs and a line joining match pairs. In step 906, if theta is greater

than a preselected maximum angle (i.e., the maximally allowed angle), then the method loops back to step 902 to examine another pair. If not, however, the method continues on to step 907 to get the stretch (i.e., compression and/or expansion transformation of one strip relative to another), which is determined from feature and corresponding match pairs as follows:

$$\%Stretch = \frac{\%F_{Dist}}{\%M_{Dist}} \qquad (6)$$

Referring to Figs. 9D-E, the operation of determining stretch between strips is graphically shown. In Fig. 9D, the right strip is expanded or stretched relative to the left strip. The exact amount of stretch may be determined by the ratio of F_{DIST} to M_{DIST}. The amount of stretching between images may not be constant, however. Therefore, as shown in Fig. 9E, compression and/or expansion is measured at several locations of the image strips. From this information, areas may be selectively corrected for stretching.

In step 908, if stretch is greater than the maximum stretch, then the method loops back to step 902. Otherwise, the method continues on to step 909 to store the pair information in a Pair_List. After step 909, the method loops back to step 902 to process additional pairs.

After all pairs have been processed (i.e., no at step 902), the method continues on to step 910 to calculate the mean and standard deviation of theta and stretch in the Pair_List. In step 911, stretch values in the Pair_List are normalized according to the obtained standard deviation of theta. In step 912, a loop is established to examine all pairs of the Pair_List. If an additional pair remains at step 912, then the method continues on to step 913 to determine the distance between a pair and population mean in (theta, stretch) plane.

In step 914, if distance is greater than the computed standard deviation, then the method loops back to step 912 to examine another pair in the Pair_List. Otherwise (no at

step 914), the pair is removed from the Pair_List at step 915, after which the method loops back to step 912.

If no more pairs remain to be examined in step 912, the method jumps to step 916 to rank the pairs in the Pair_List. This ranking is based on the closeness of a pair to the center of the population and correlation index. In step 917, the two features and matches which comprise the best Pair_List are returned, and the method concludes.

Referring now to Figs. 10A-C, the Calculate Dimensions method 1000, which is invoked in step 712, is illustrated. For two features, their associated matches, and the bounding rectangles of two image strips (first strip and second strip), the method calculates the dimensions of the merged image. The method assumes that Automerge corrects only for rotation, stretching, and translation transformations. Those skilled in the art will appreciate that other transformations may be accommodated within the scope of the present invention. The steps of the method 1000 will now be described.

In step 1001, the coordinates of two points in strip one, which, together with the two features form a rectangle of arbitrary width (Rect1 of Fig. 10B) are calculated. In step 1002, a corresponding rectangle (Rect2 of Fig. 10B) is calculated for strip 2. In step 1003, for the two rectangles (Rect1 and Rect2) a projective affine transformation matrix is determined.

In step 1004, a third rectangle (Rect3 of Fig. 10C) is calculated from an outline of strip 2. In step 1005, a fourth rectangle (Rect4 of Fig. 10C) is calculated from a bounding rectangle of Rect3. In step 1006 the biggest rectangle, derived from Rect4 and the bounding rectangle of strip 1, is returned.

Referring now to Fig. 11, the M_MergeImage method 1100 is illustrated. For two image strips (strip 1 and strip 2), two features in strip 1, and two matches in strip 2, the method combines strip 1 and strip 2 into a resulting image (Result).

The individual steps of the method will now be described. In step 1101, a first rectangle (Rect1) is calculated for strip 1 (as described hereinabove). In step 1102, a second rectangle (Rect2) is calculated for strip 2. In step 1103, from the two rectangles (Rect1 and Rect2) a projective affine transformation matrix (shown in Fig. 10B) is calculated. In step 1104, a third rectangle (Rect3) is calculated from an outline of strip 2 in Result. In step 1105, **X** and **Y** offsets of strip 1 (in Result) are calculated. In step 1106, the first strip is copied into Result. In step 1107, a projective affine transformation matrix is calculated from the two rectangles (Rect1 and Rect2).

In Step 1108, a loop is established to examine each pixel of Result. If an additional pixel exists at step 1108, then in step 1109 the position of the pixel in the second strip is calculated (based on brightness). In step 1110, the Brightness (**X**, **Y**) is replaced with bi-linear interpolation, a known technique, after which the method loops back to step 1108 for another pixel of Result. If there are no more pixels (no at step 1108 no), then the method concludes. At the conclusion of the method, the two image strips have been merged.

While the invention is described in some detail with specific reference to a single preferred embodiment and certain alternatives, there is no intent to limit the invention to that particular embodiment or those specific alternatives. While specific examples have been shown using monochromatic bitmapped graphic images, for example, it will be apparent to those skilled in the art to apply the teachings of the present invention to other formats, including color bitmapped images, vector images, and the like. Therefore, the

true scope of the invention is defined not by the foregoing description but by the following claims.

WHAT IS CLAIMED IS:

1. A method for automatically merging first and second images stored on a computer system, the method comprising:

(a) locating at least one feature in the first image strip;

(b) matching a feature in the second image strip which is substantially similar to said at least one feature in the first image; and

(c) merging the two image strips into a single image by aligning the two features.

2. The method of claim 1, wherein before step (c) further comprises:

determining a geometry of the first strip;

determining a geometry of the second strip; and

transforming the geometry of the second strip to that of the geometry of the first strip.

3. The method of claim 2, wherein said transforming step includes translation, rotation, compression, and expansion transformations.

4. The method of claim 1, wherein said image strips are bitmap images.

5. The method of claim 1, wherein said image strips are vector-format images.

6. In a system for providing images to a digital computer, the improvement comprising:

means for acquiring a source image as a plurality of image strips;

means for matching features between at least two of the image strips: and

means for merging said at least two image strips into a single image by aligning the matching features.

7. The system of claim 6, wherein said acquiring means is a handheld scanner.

8. The system of claim 6, further comprising:

means for transforming the geometry of the second strip to that of the first strip.

9. A system for entering images into a computer, the system comprising:

a computer having a memory and a processor;

scanner means for acquiring a source image as a plurality of image strips;

means for locating features common to at least two of the image strips; and

means for merging said at least two image strips by aligning the common features.

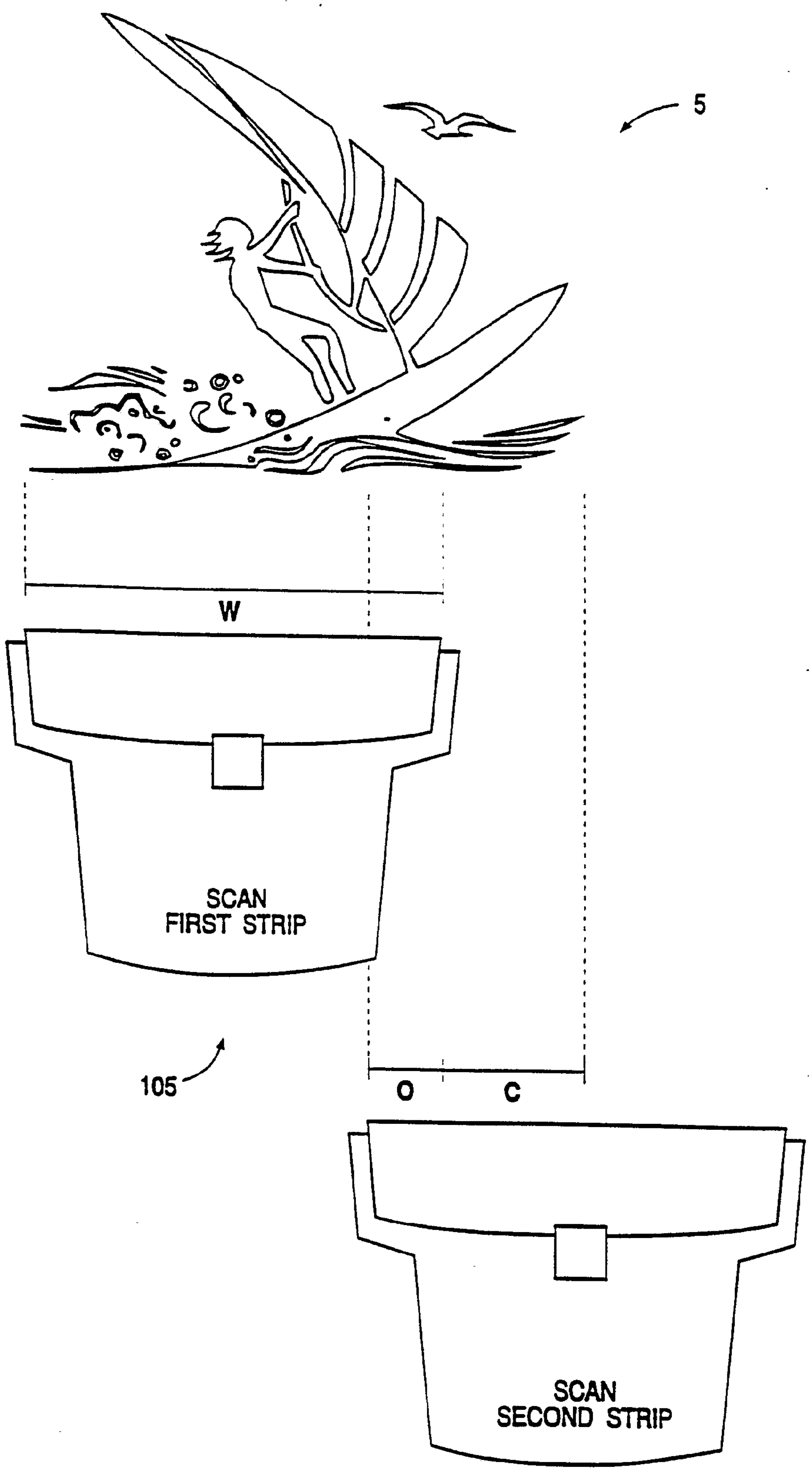


FIG. 1A

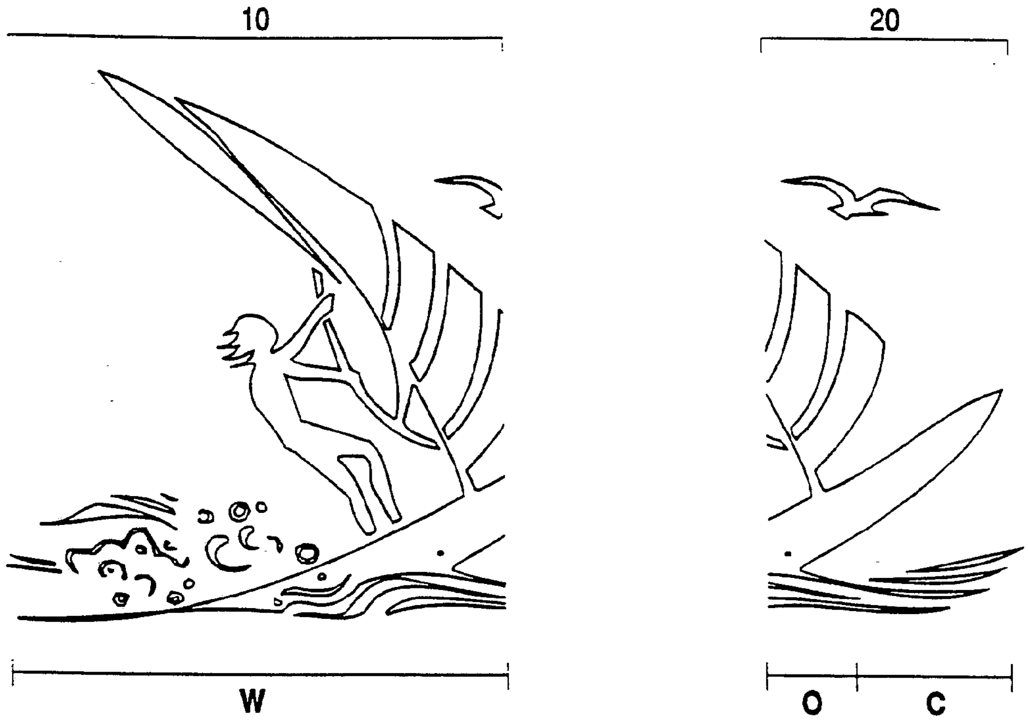


FIG. 1B

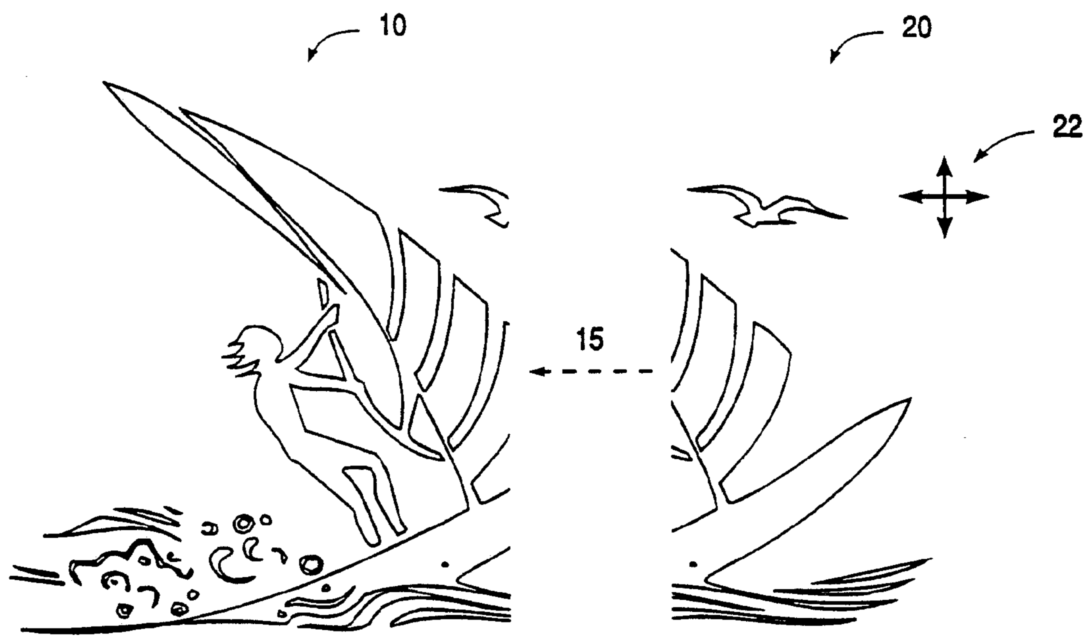


FIG. 1C

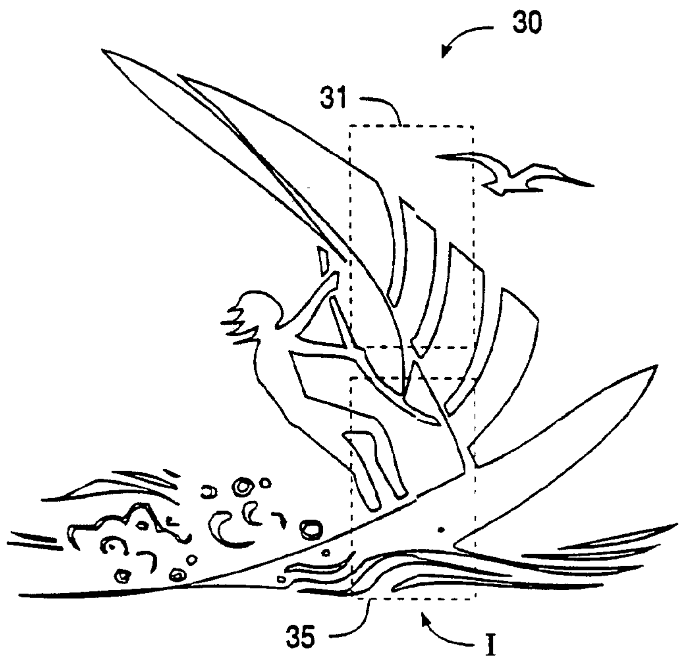


FIG. 1D

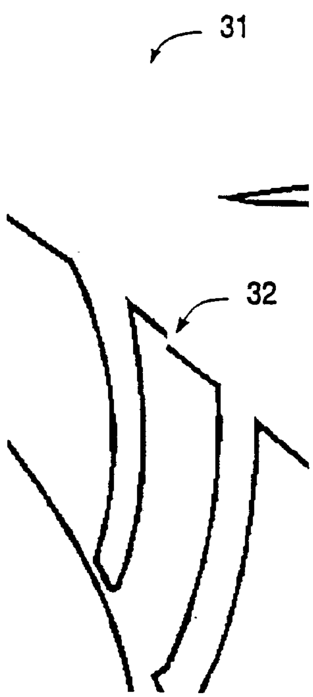


FIG. 1E

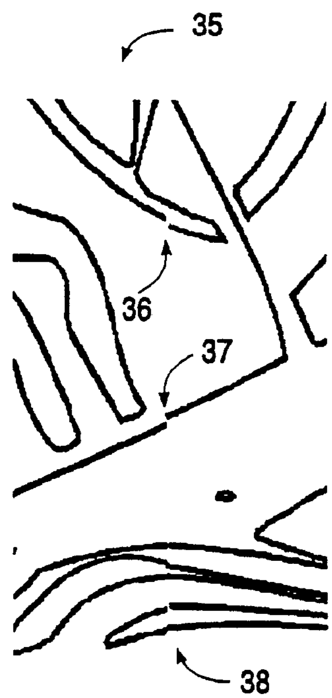


FIG. 1F

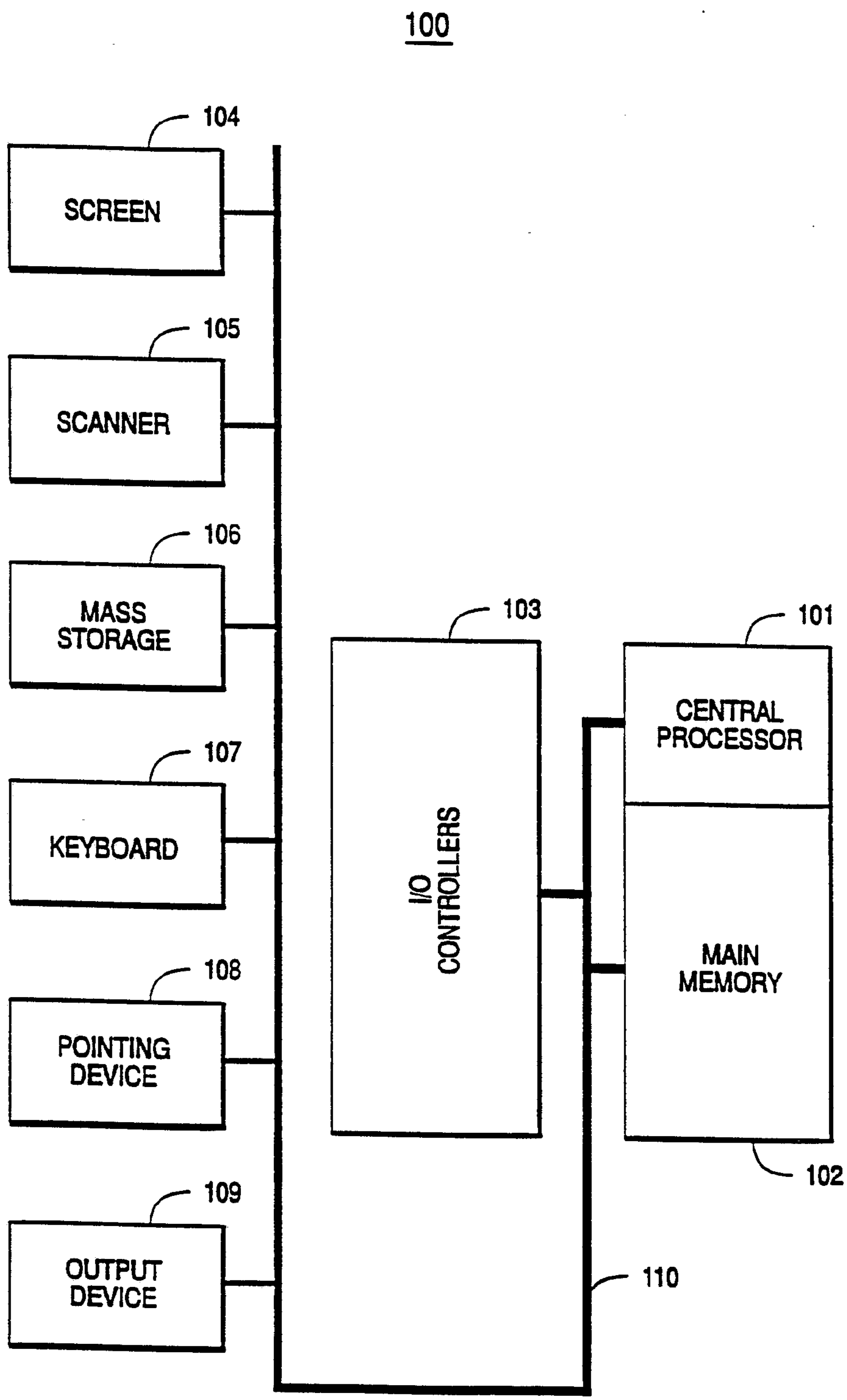

100
104
SCREEN
105
SCANNER
106
MASS STORAGE
107
KEYBOARD
108
POINTING DEVICE
109
OUTPUT DEVICE
103
I/O CONTROLLERS
101
CENTRAL PROCESSOR
MAIN MEMORY
102
110

FIG. 2

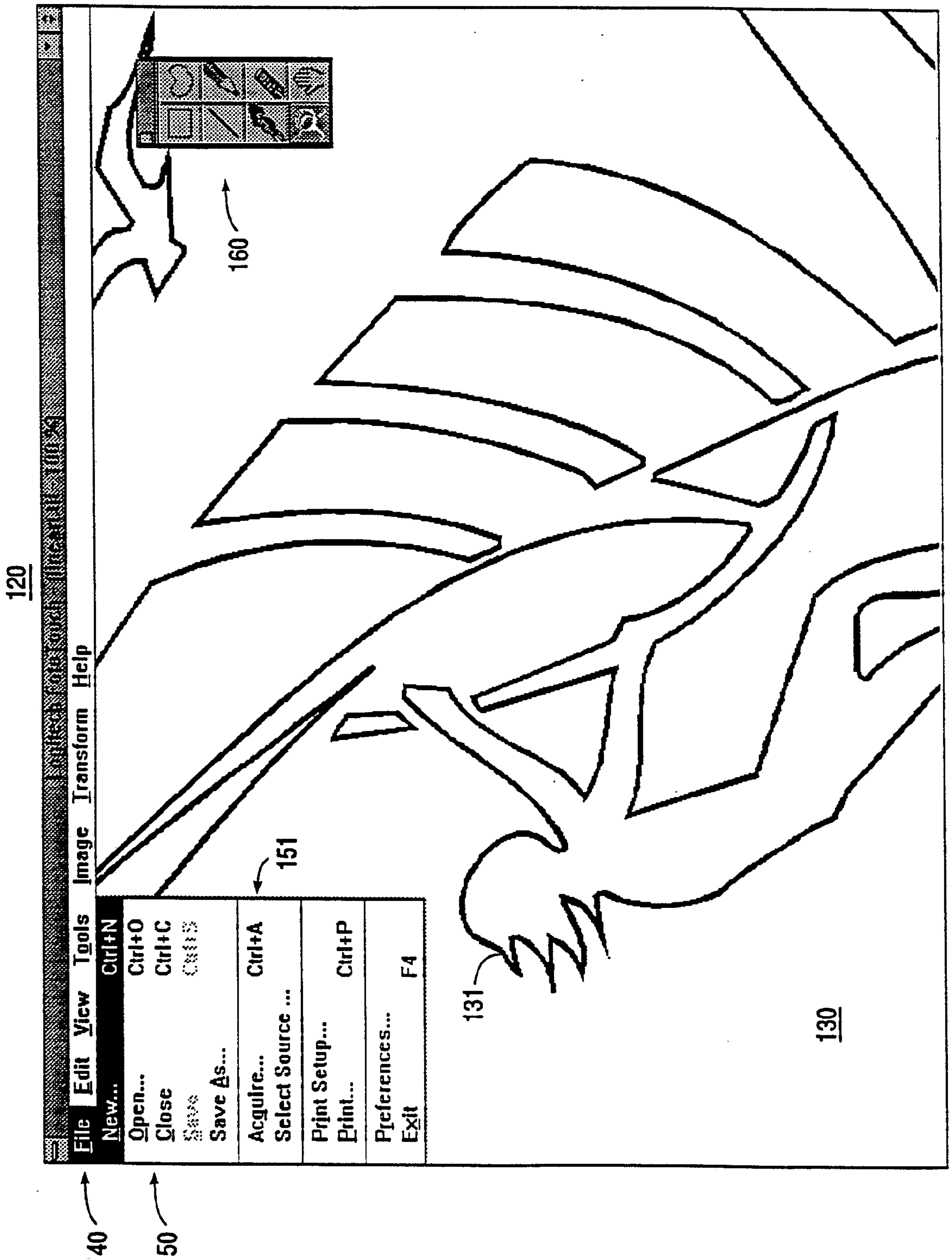

120
File Edit View Tools Image Transform Help
New... Ctrl+N
Open... Ctrl+O
Close Ctrl+C
Save Ctrl+S
Save As...
Acquire... Ctrl+A
Select Source ...
Print Setup...
Print... Ctrl+P
Preferences...
Exit F4
140
150
151
160
131
130

FIG. 3A

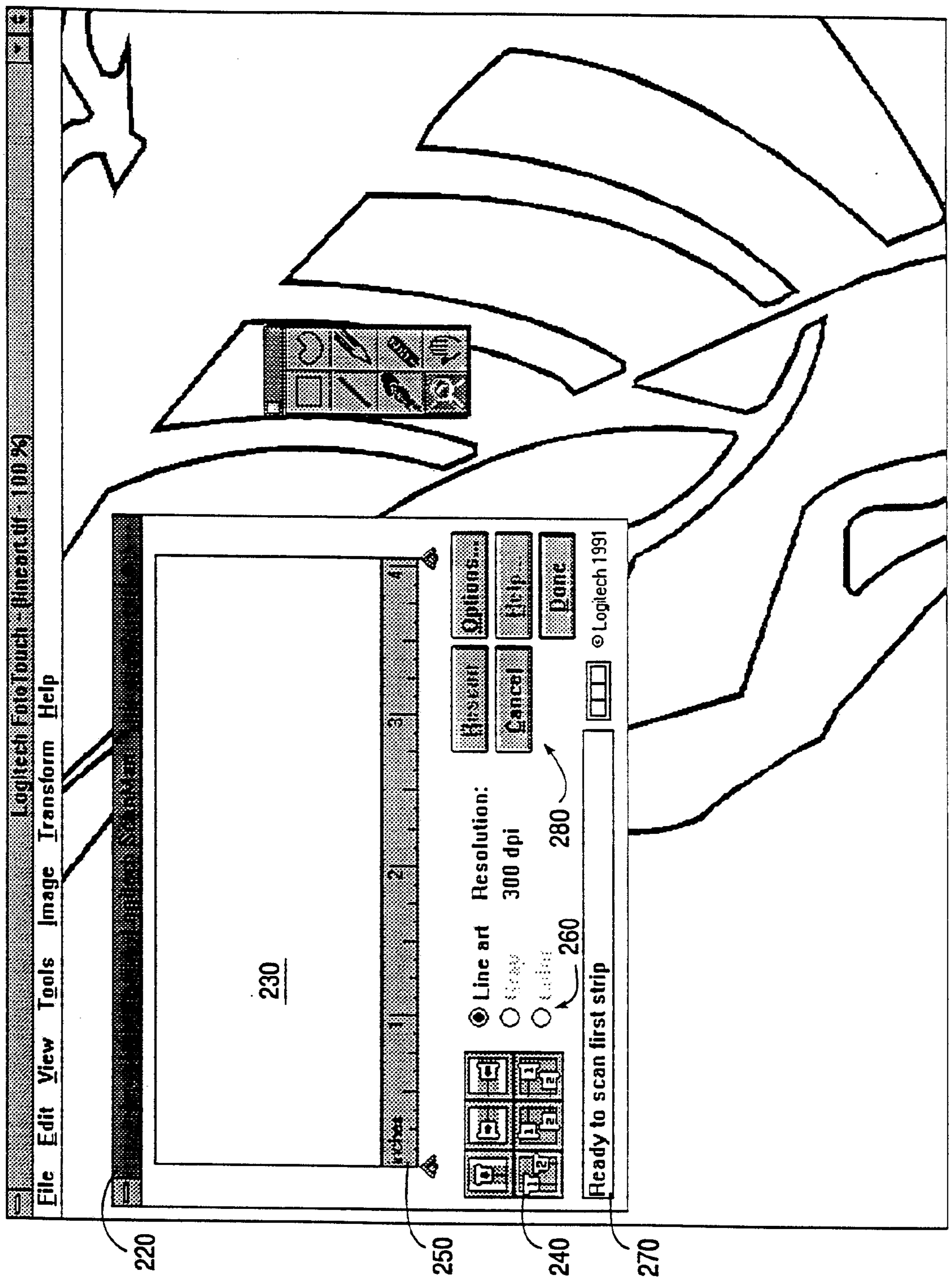

File Edit View Tools Image Transform Help
230
Line art
Resolution:
300 dpi
Rescan
Cancel
Options...
Done
© Logitech 1991
Ready to scan first strip
220
250
240
270
260
280

FIG. 3B

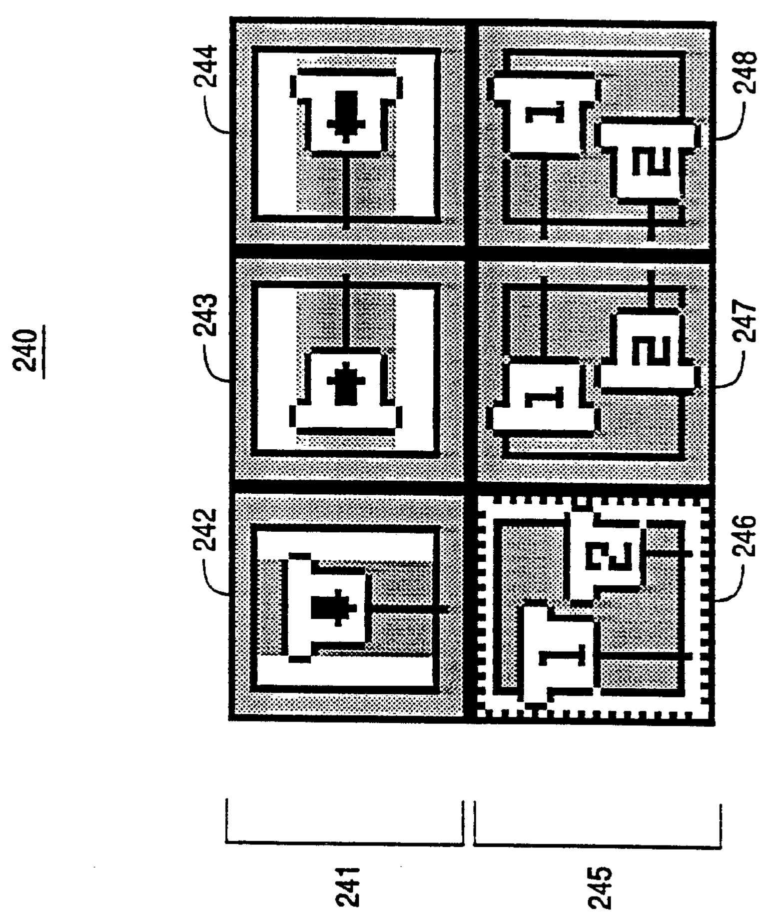


FIG. 3C

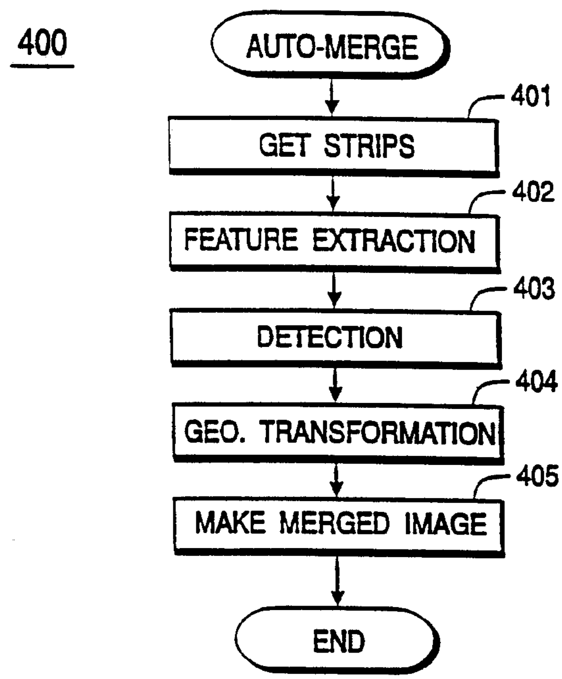

400
AUTO-MERGE
401
GET STRIPS
402
FEATURE EXTRACTION
403
DETECTION
404
GEO. TRANSFORMATION
405
MAKE MERGED IMAGE
END

FIG. 4

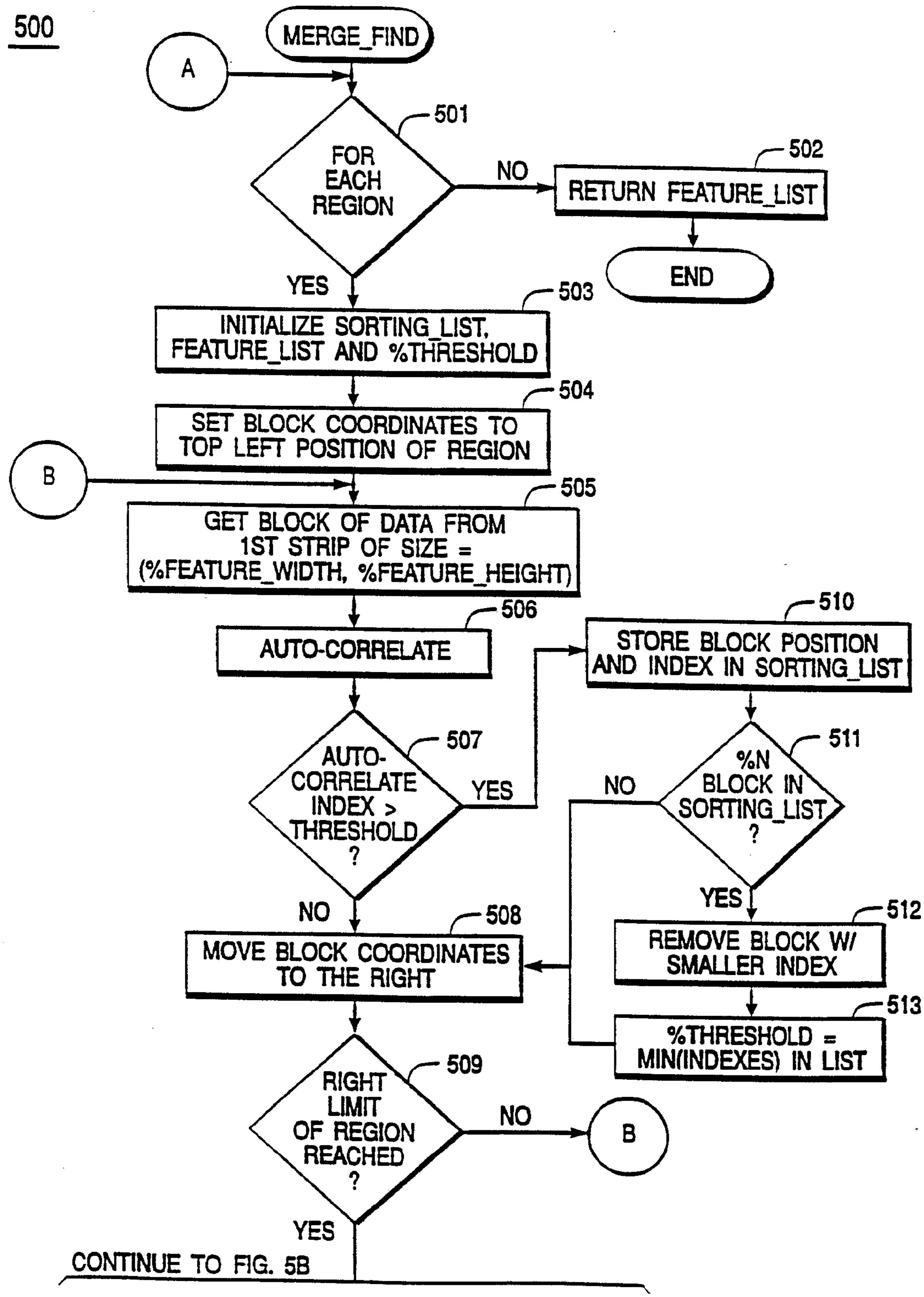

500
MERGE_FIND
A
FOR EACH REGION
501
NO
RETURN FEATURE_LIST
502
END
YES
INITIALIZE SORTING_LIST, FEATURE_LIST AND %THRESHOLD
503
SET BLOCK COORDINATES TO TOP LEFT POSITION OF REGION
504
B
GET BLOCK OF DATA FROM 1ST STRIP OF SIZE = (%FEATURE_WIDTH, %FEATURE_HEIGHT)
505
AUTO-CORRELATE
506
STORE BLOCK POSITION AND INDEX IN SORTING_LIST
510
AUTO-CORRELATE INDEX > THRESHOLD ?
507
YES
NO
%N BLOCK IN SORTING_LIST ?
511
YES
REMOVE BLOCK W/ SMALLER INDEX
512
NO
MOVE BLOCK COORDINATES TO THE RIGHT
508
%THRESHOLD = MIN(INDEXES) IN LIST
513
RIGHT LIMIT OF REGION REACHED ?
509
NO
B
YES
CONTINUE TO FIG. 5B

FIG. 5A

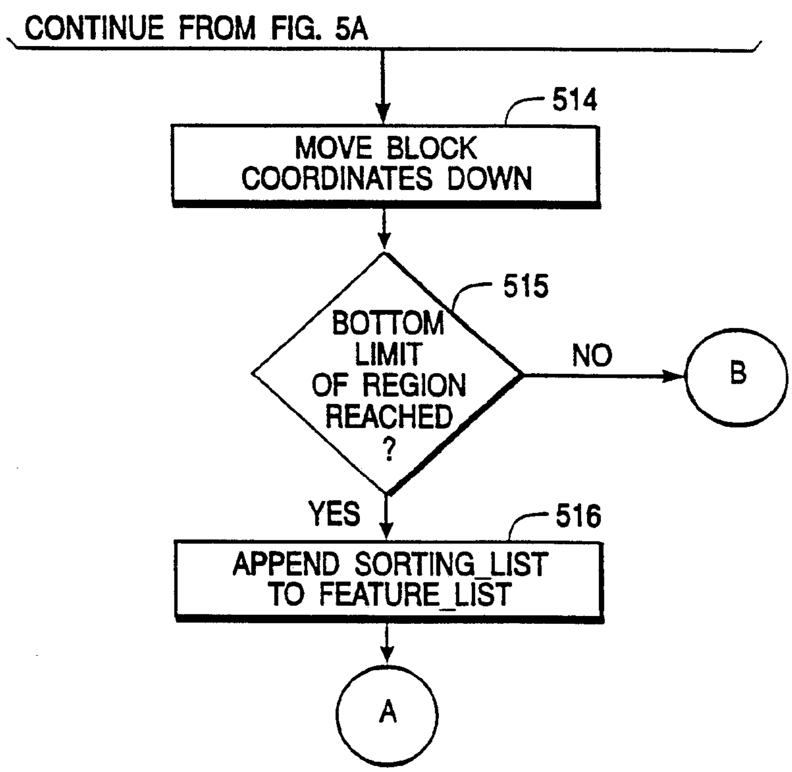

CONTINUE FROM FIG. 5A
514
MOVE BLOCK COORDINATES DOWN
515
BOTTOM LIMIT OF REGION REACHED ?
NO
B
YES
516
APPEND SORTING_LIST TO FEATURE_LIST
A

FIG. 5B

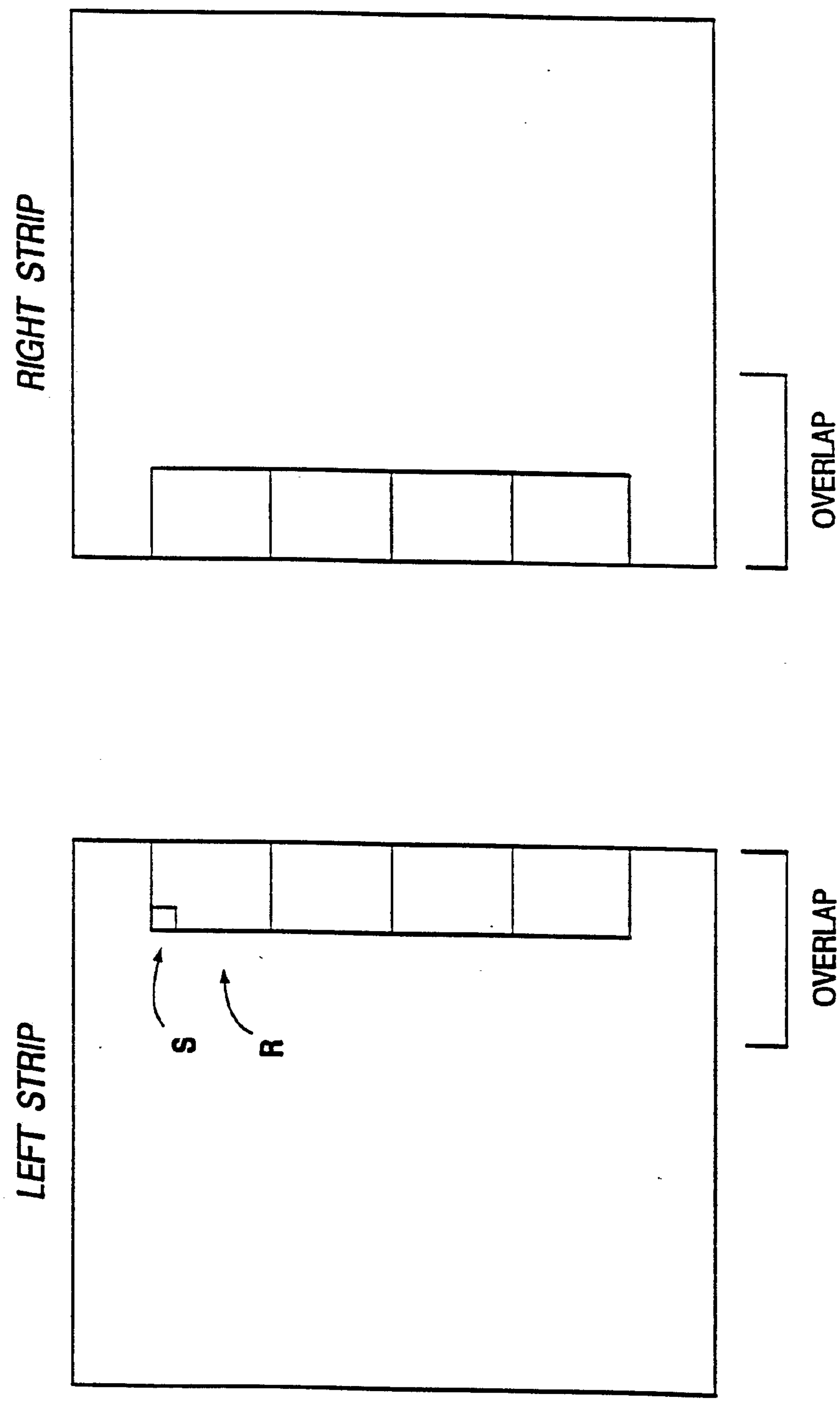


FIG. 5C''

FIG. 5C'

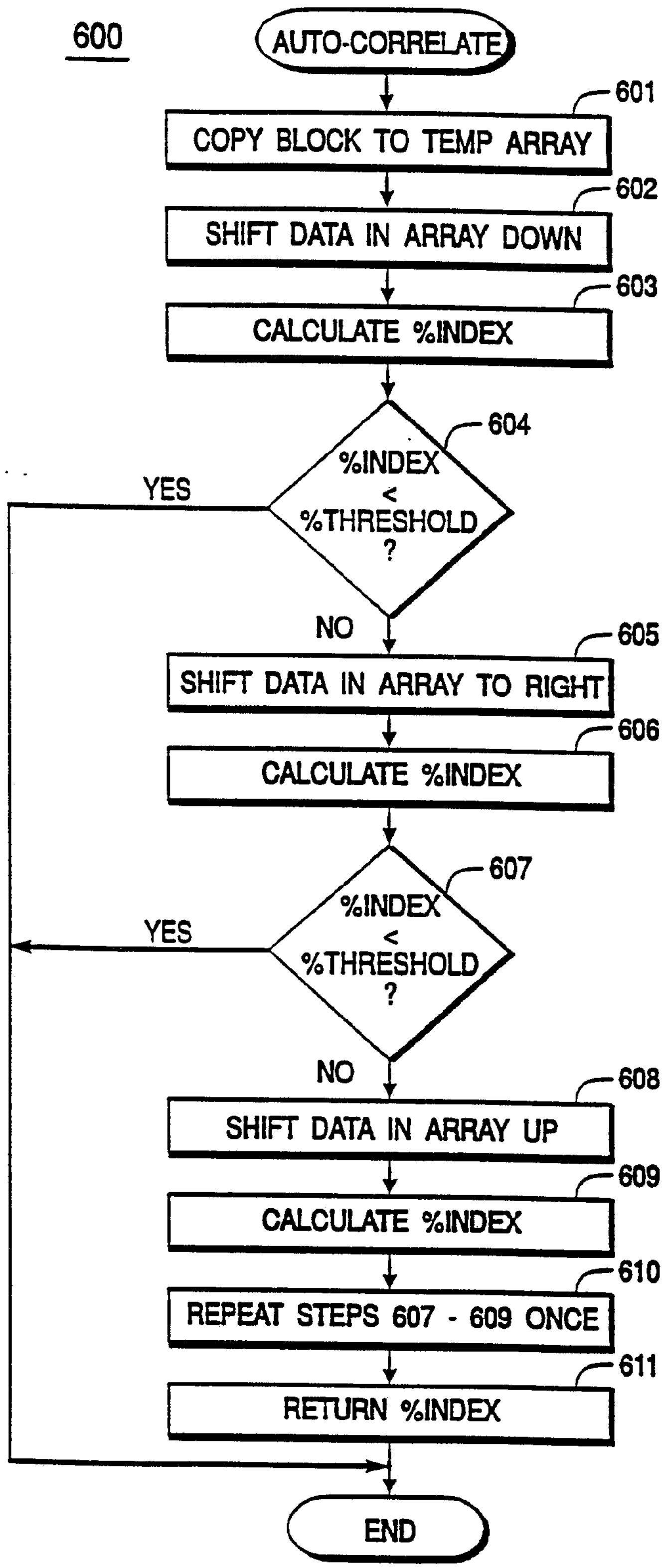

600
AUTO-CORRELATE
601
COPY BLOCK TO TEMP ARRAY
602
SHIFT DATA IN ARRAY DOWN
603
CALCULATE %INDEX
604
%INDEX < %THRESHOLD ?
YES
NO
605
SHIFT DATA IN ARRAY TO RIGHT
606
CALCULATE %INDEX
607
%INDEX < %THRESHOLD ?
YES
NO
608
SHIFT DATA IN ARRAY UP
609
CALCULATE %INDEX
610
REPEAT STEPS 607 - 609 ONCE
611
RETURN %INDEX
END

FIG. 6A

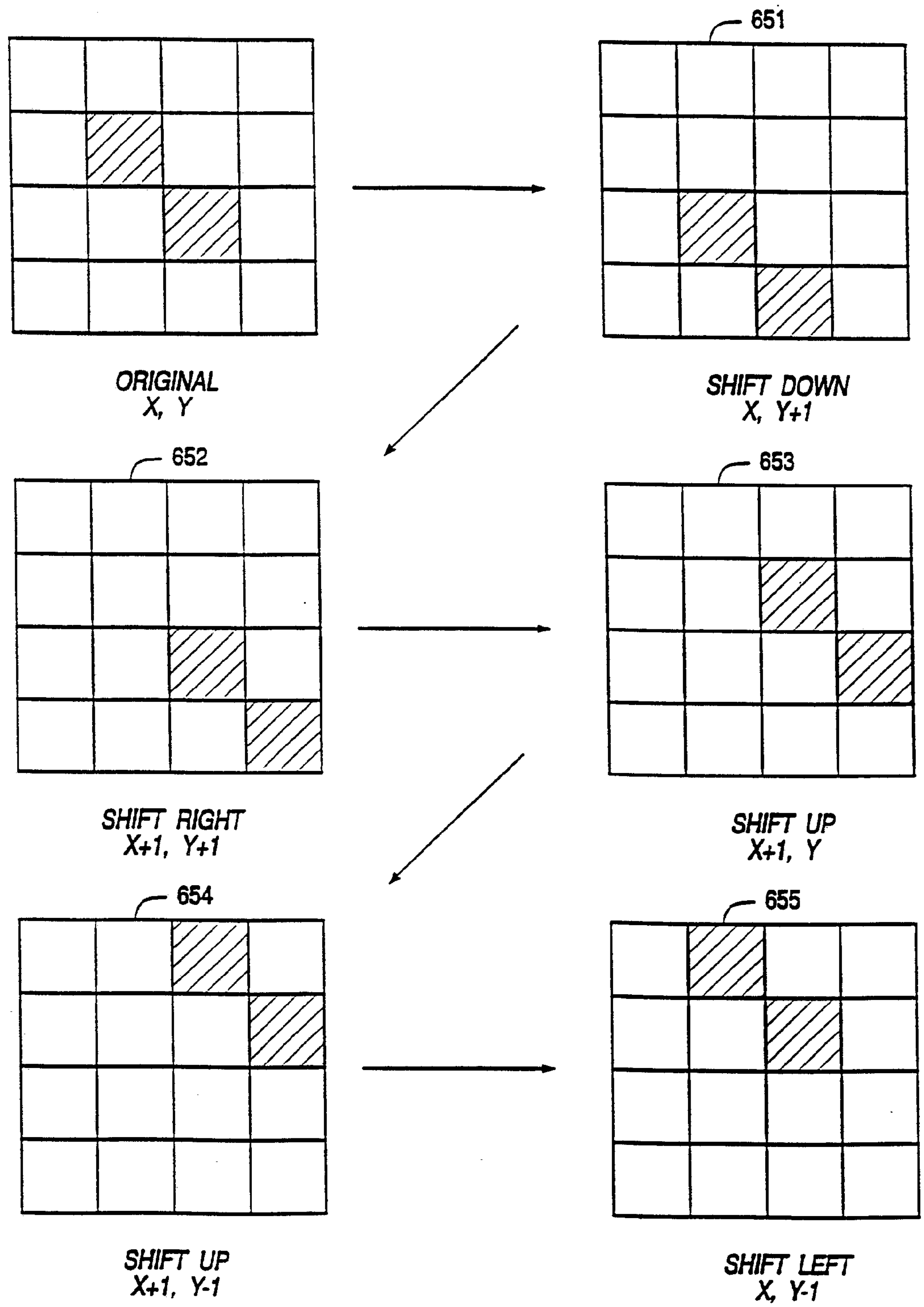

651
ORIGINAL
X, Y
SHIFT DOWN
X, Y+1
652
653
SHIFT RIGHT
X+1, Y+1
SHIFT UP
X+1, Y
654
655
SHIFT UP
X+1, Y-1
SHIFT LEFT
X, Y-1

FIG. 6B

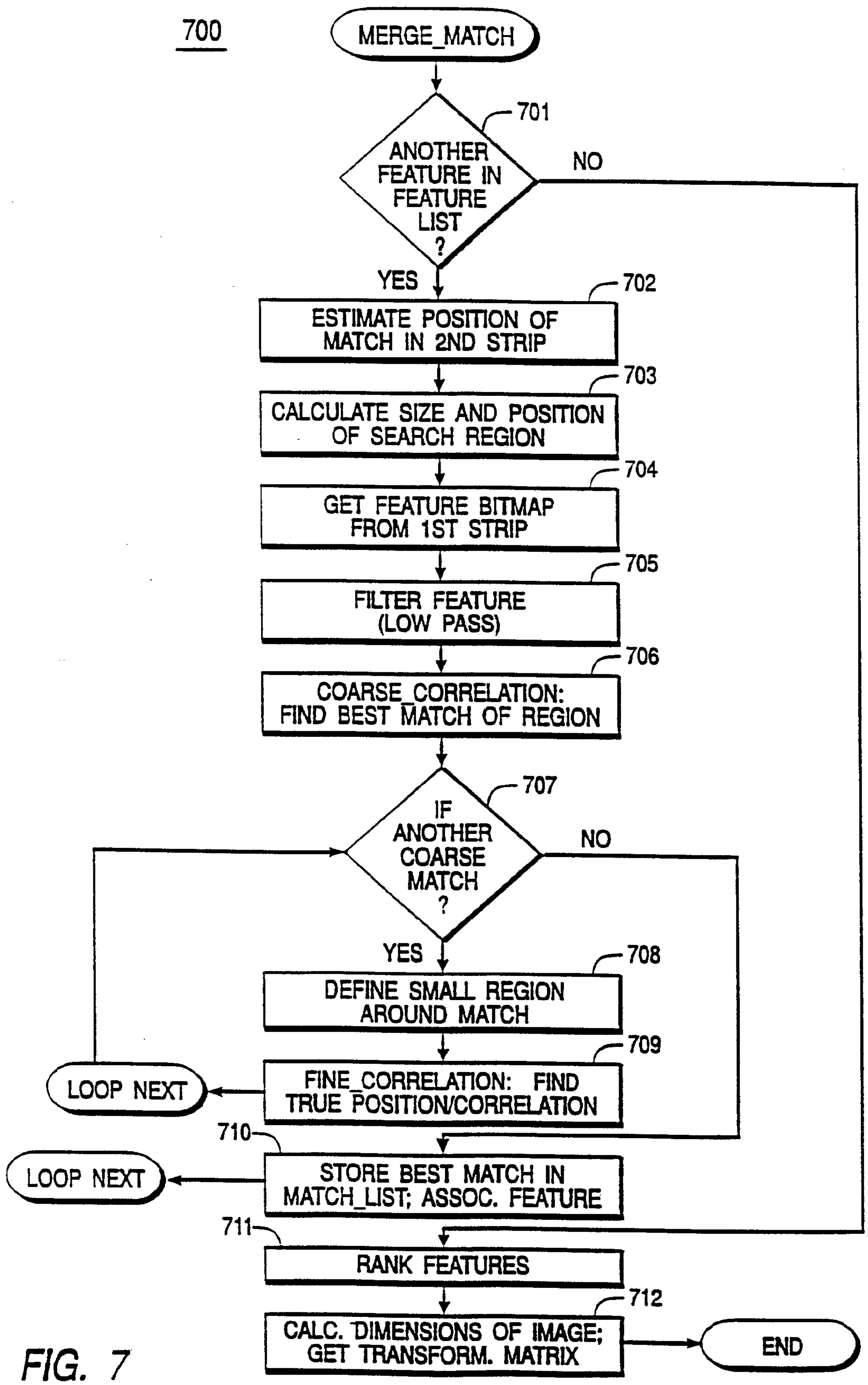

700
MERGE_MATCH
701
ANOTHER FEATURE IN FEATURE LIST ?
NO
YES
702
ESTIMATE POSITION OF MATCH IN 2ND STRIP
703
CALCULATE SIZE AND POSITION OF SEARCH REGION
704
GET FEATURE BITMAP FROM 1ST STRIP
705
FILTER FEATURE (LOW PASS)
706
COARSE_CORRELATION: FIND BEST MATCH OF REGION
707
IF ANOTHER COARSE MATCH ?
NO
YES
708
DEFINE SMALL REGION AROUND MATCH
709
FINE_CORRELATION: FIND TRUE POSITION/CORRELATION
LOOP NEXT
710
STORE BEST MATCH IN MATCH_LIST; ASSOC. FEATURE
LOOP NEXT
711
RANK FEATURES
712
CALC. DIMENSIONS OF IMAGE; GET TRANSFORM. MATRIX
END

FIG. 7

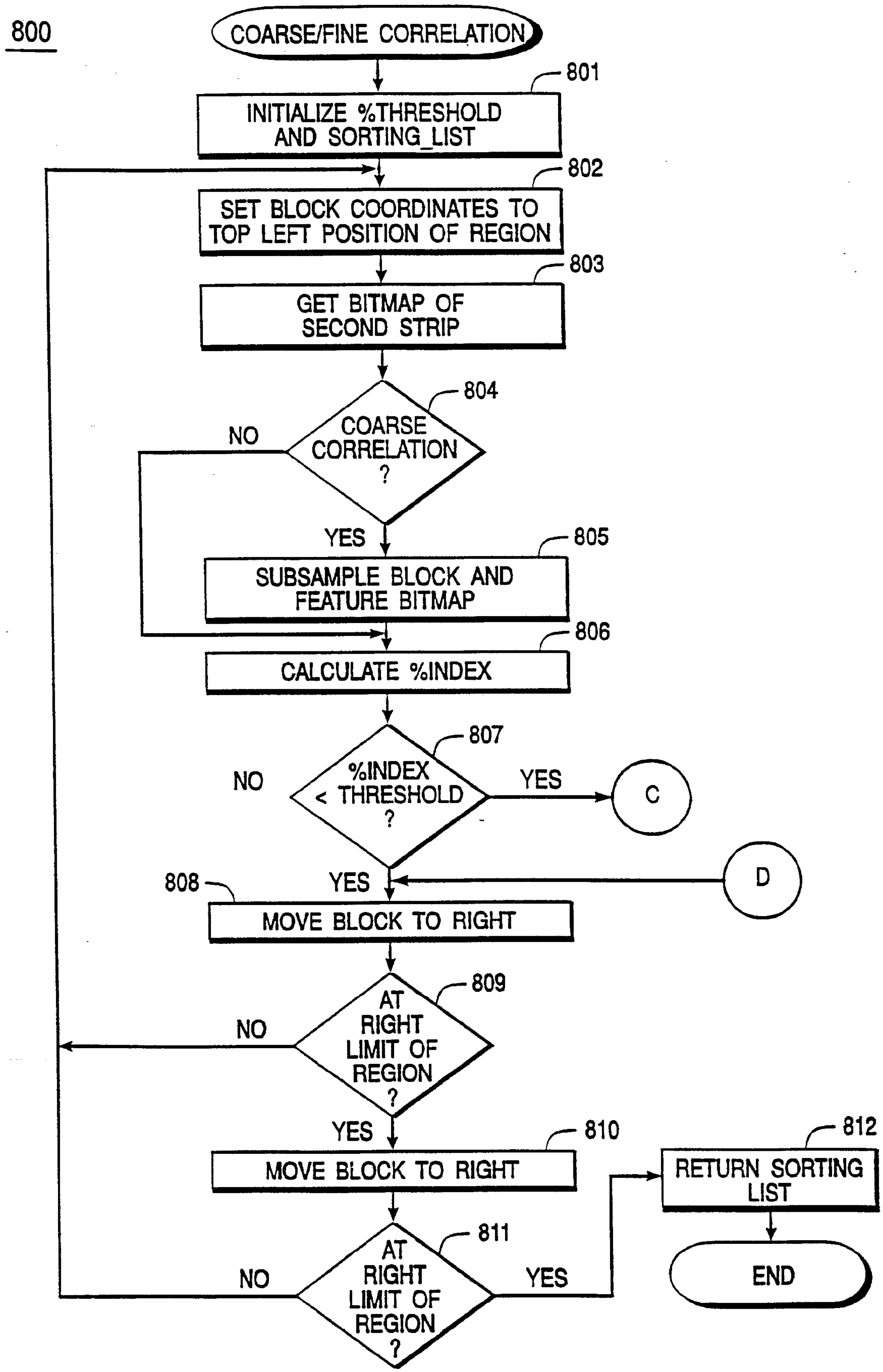

800
COARSE/FINE CORRELATION
801
INITIALIZE %THRESHOLD AND SORTING_LIST
802
SET BLOCK COORDINATES TO TOP LEFT POSITION OF REGION
803
GET BITMAP OF SECOND STRIP
804
COARSE CORRELATION ?
NO
YES
805
SUBSAMPLE BLOCK AND FEATURE BITMAP
806
CALCULATE %INDEX
807
%INDEX < THRESHOLD ?
NO
YES
C
D
808
YES
MOVE BLOCK TO RIGHT
809
AT RIGHT LIMIT OF REGION ?
NO
YES
810
MOVE BLOCK TO RIGHT
811
AT RIGHT LIMIT OF REGION ?
NO
YES
812
RETURN SORTING LIST
END

FIG. 8A

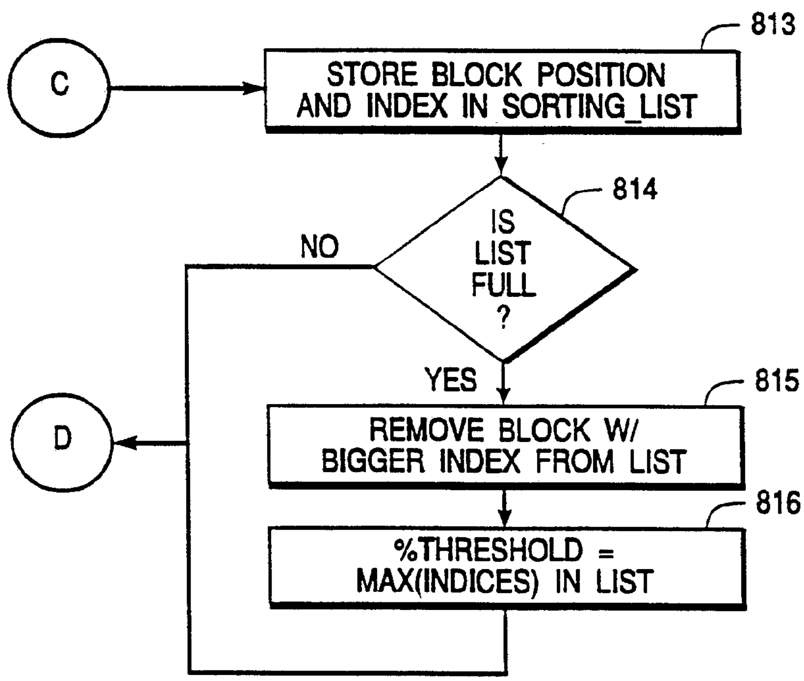

813
C
STORE BLOCK POSITION AND INDEX IN SORTING_LIST
814
IS LIST FULL ?
NO
YES
815
D
REMOVE BLOCK W/ BIGGER INDEX FROM LIST
816
%THRESHOLD = MAX(INDICES) IN LIST

FIG. 8B

900

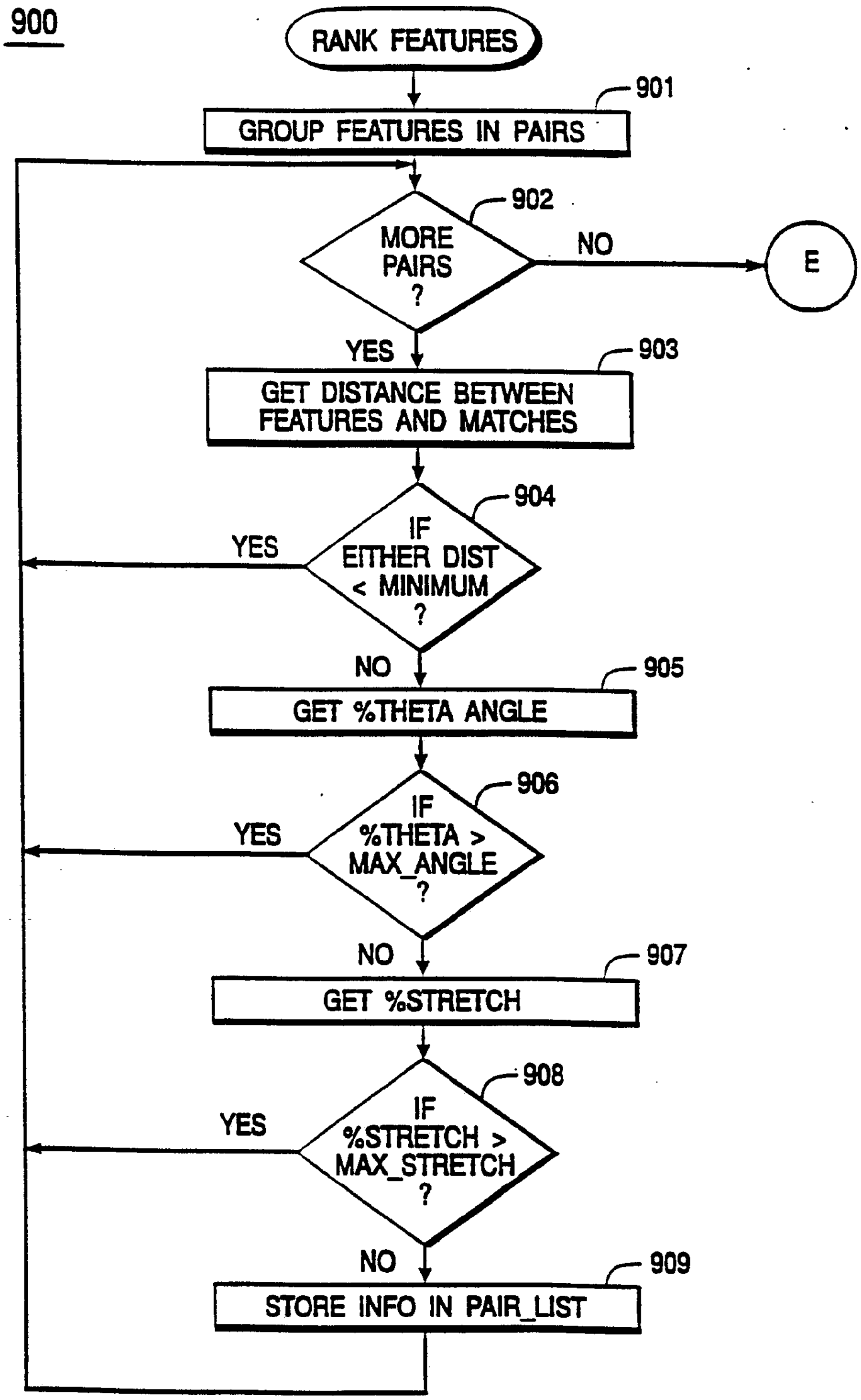

RANK FEATURES
901
GROUP FEATURES IN PAIRS
902
MORE PAIRS ?
NO
E
YES
903
GET DISTANCE BETWEEN FEATURES AND MATCHES
904
IF EITHER DIST < MINIMUM ?
YES
NO
905
GET %THETA ANGLE
906
IF %THETA > MAX_ANGLE ?
YES
NO
907
GET %STRETCH
908
IF %STRETCH > MAX_STRETCH ?
YES
NO
909
STORE INFO IN PAIR_LIST

FIG. 9A

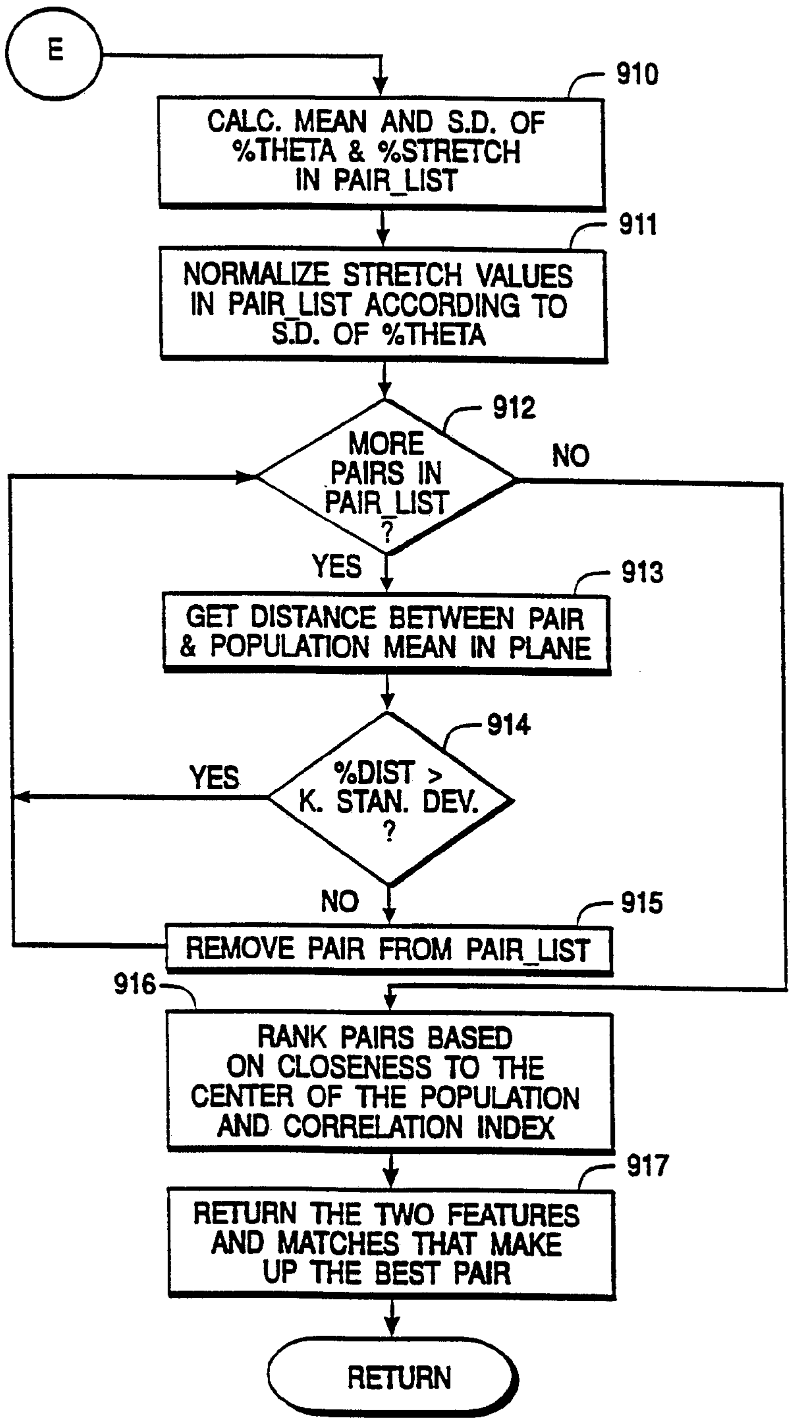
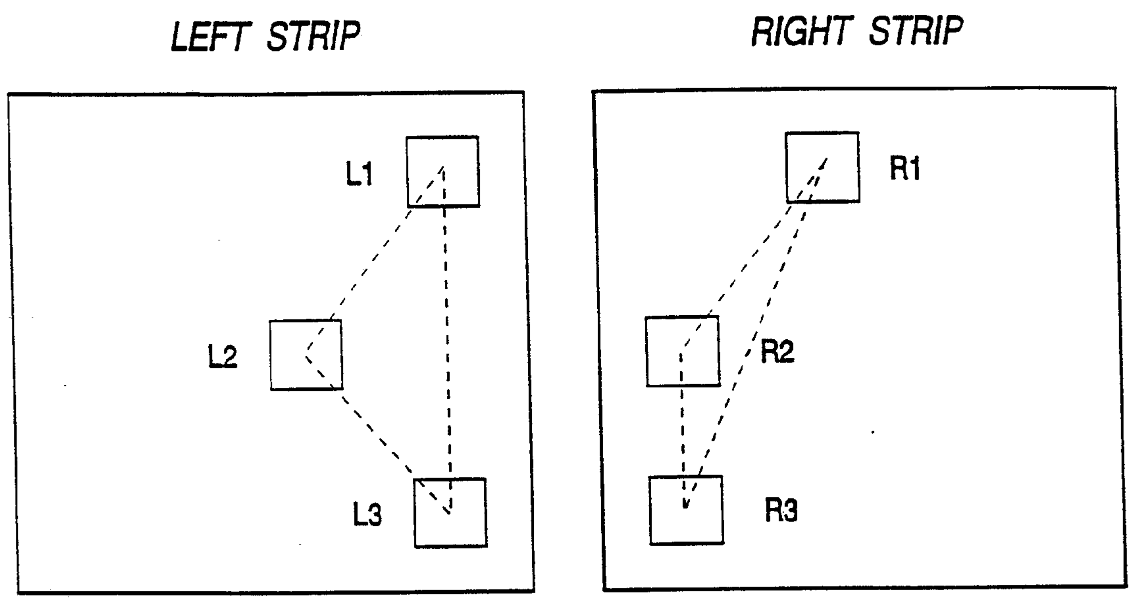

E
910
CALC. MEAN AND S.D. OF %THETA & %STRETCH IN PAIR_LIST
911
NORMALIZE STRETCH VALUES IN PAIR_LIST ACCORDING TO S.D. OF %THETA
912
MORE PAIRS IN PAIR_LIST ?
NO
YES
913
GET DISTANCE BETWEEN PAIR & POPULATION MEAN IN PLANE
914
%DIST > K. STAN. DEV. ?
YES
NO
915
REMOVE PAIR FROM PAIR_LIST
916
RANK PAIRS BASED ON CLOSENESS TO THE CENTER OF THE POPULATION AND CORRELATION INDEX
917
RETURN THE TWO FEATURES AND MATCHES THAT MAKE UP THE BEST PAIR
RETURN

FIG. 9B



PAIRS	DIST (L)	DIST (R)	ROTATE	STRETCH
L1, L2	D(L1, L2)	D(R1, R2)	θ_1	$\%S_1$
L1, L3	D(L1, L2)	D(R1, R2)	θ_2	$\%S_2$
L2, L3	D(L1, L2)	D(R1, R2)	θ_3	$\%S_3$

FIG. 9C

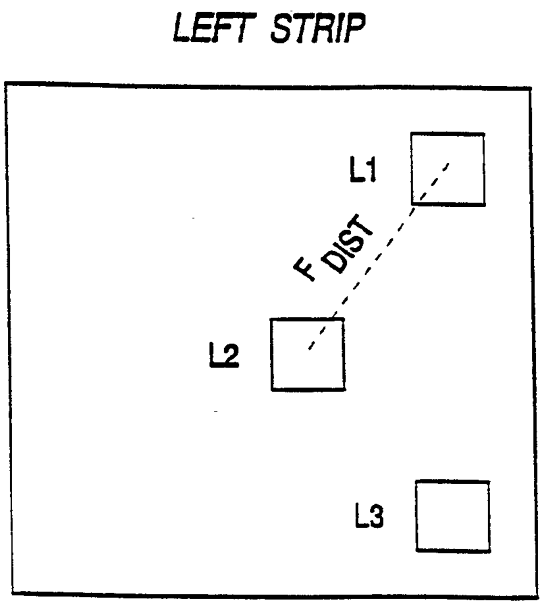


FIG. 9D′

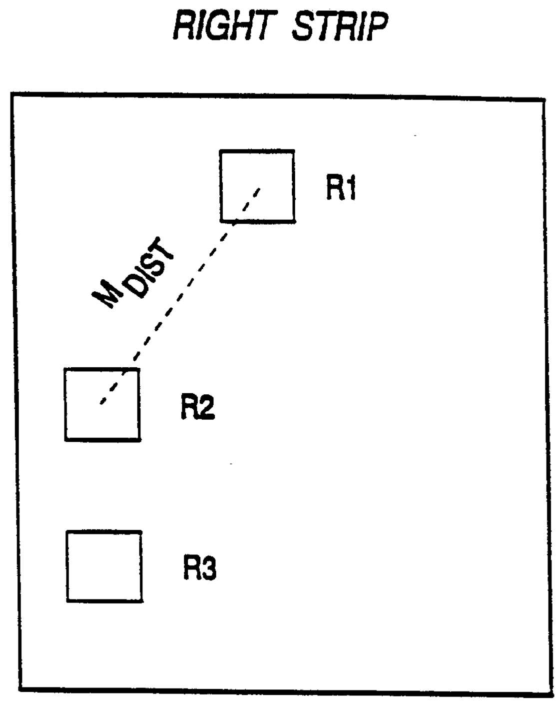


FIG. 9D″

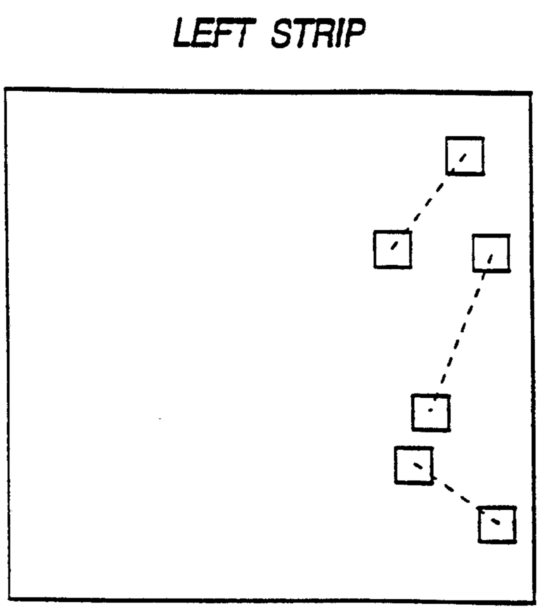


FIG. 9E′

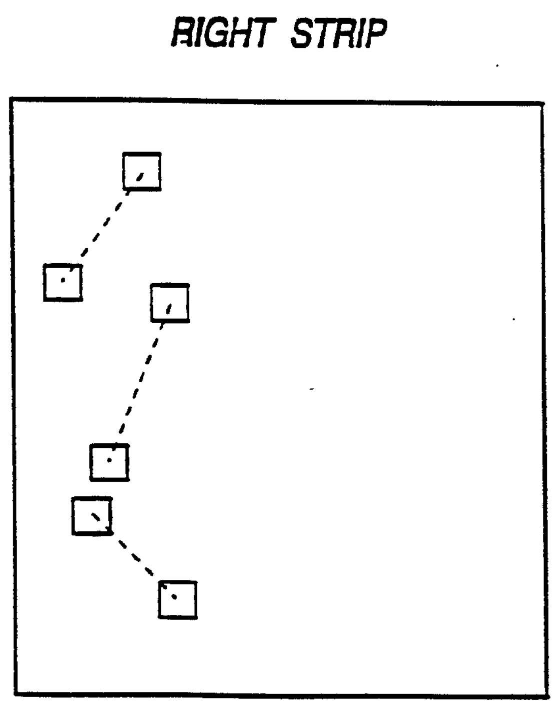


FIG. 9E″

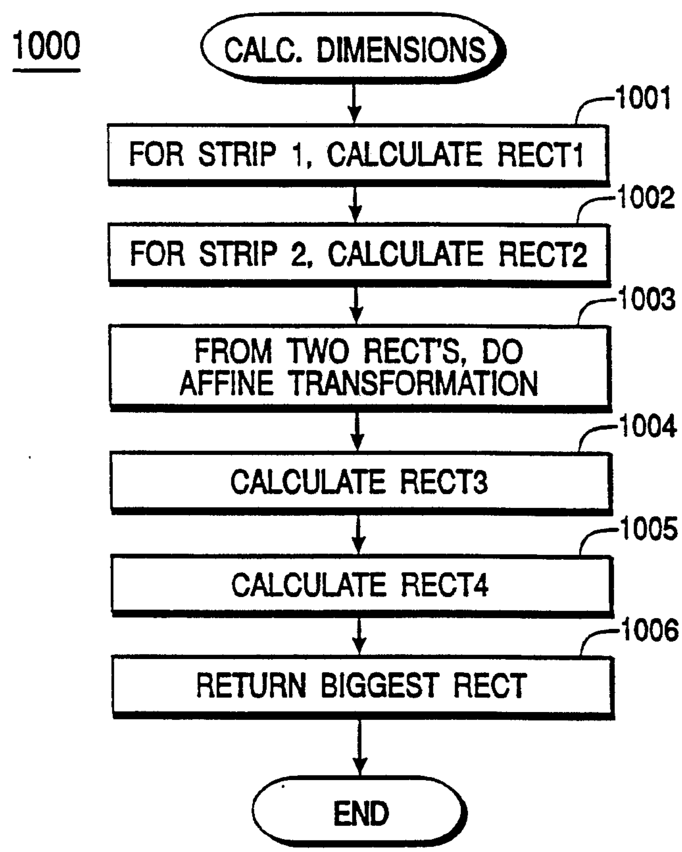

1000
CALC. DIMENSIONS
1001
FOR STRIP 1, CALCULATE RECT1
1002
FOR STRIP 2, CALCULATE RECT2
1003
FROM TWO RECT'S, DO AFFINE TRANSFORMATION
1004
CALCULATE RECT3
1005
CALCULATE RECT4
1006
RETURN BIGGEST RECT
END

FIG. 10A

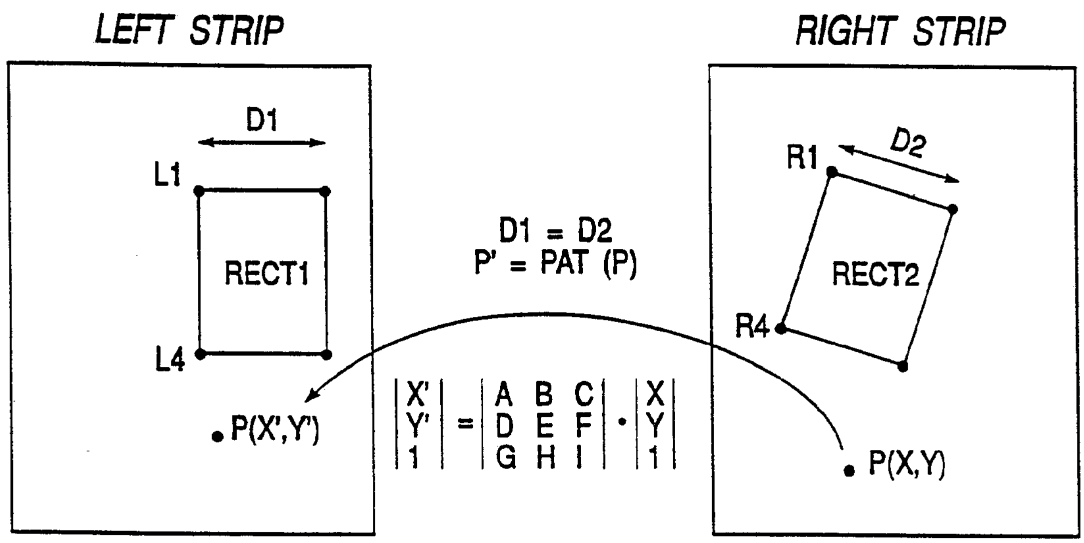


FIG. 10B

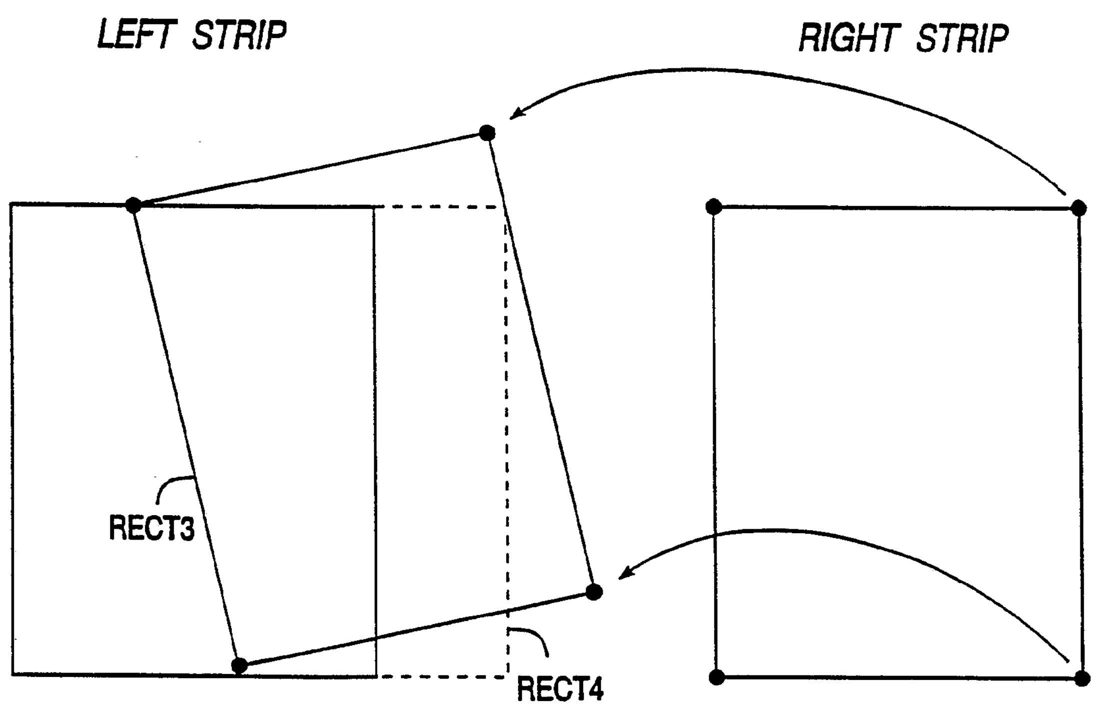


FIG. 10C

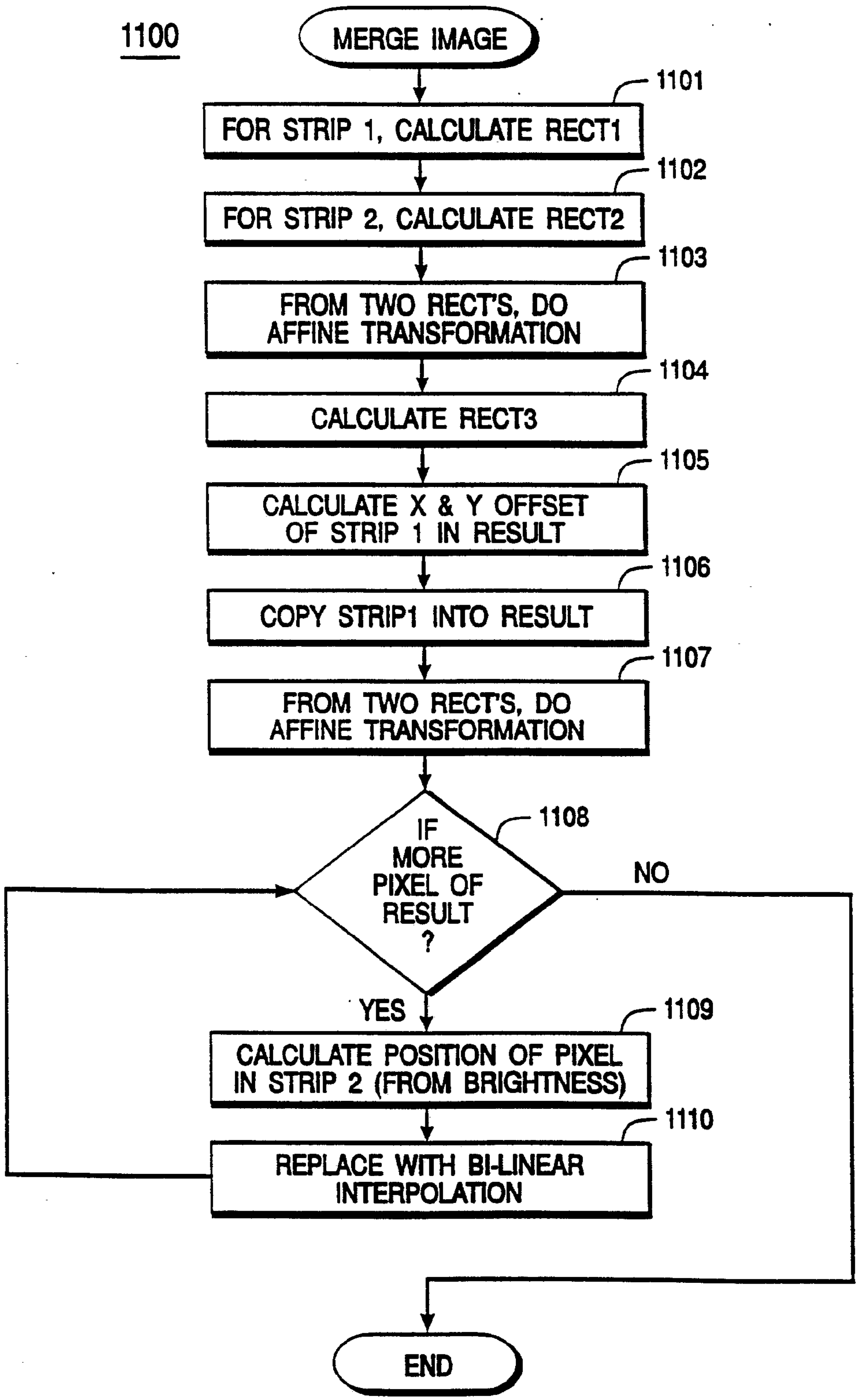

1100
MERGE IMAGE
1101
FOR STRIP 1, CALCULATE RECT1
1102
FOR STRIP 2, CALCULATE RECT2
1103
FROM TWO RECT'S, DO AFFINE TRANSFORMATION
1104
CALCULATE RECT3
1105
CALCULATE X & Y OFFSET OF STRIP 1 IN RESULT
1106
COPY STRIP1 INTO RESULT
1107
FROM TWO RECT'S, DO AFFINE TRANSFORMATION
1108
IF MORE PIXEL OF RESULT ?
NO
YES
1109
CALCULATE POSITION OF PIXEL IN STRIP 2 (FROM BRIGHTNESS)
1110
REPLACE WITH BI-LINEAR INTERPOLATION
END

FIG. 11

INTERNATIONAL SEARCH REPORT

PCT/US92/10560

A. CLASSIFICATION OF SUBJECT MATTER

IPC(5) :G06F 15/62
US CL :395/135
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/125,126,127,128,129,133,134,136,137,138,139,161;382/44,46,61

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US, A, 4,272,756 (KAKUMOTO ET AL.) 09 June 1981, See col. 5, lines 55-68 and figures 1 and 7, See figure 1, item 22; col. 1, lines 5-64; col. 1, lines 5-46; figure 2, and col. 1, lines 5-46.	1,4,6,7,9
Y	US, A, 3,852,573 (DOLCH) 03 December 1974, See column 1.	2-3,5,8

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be part of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
09 FEBRUARY 1993	04 MAR 1993

Name and mailing address of the ISA/US	Authorized officer
Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	fr ALMIS JANKUS Ngoc-Ho Nguyen NGUYEN NGOC-HO INTERNATIONAL DIVISION
Facsimile No. NOT APPLICABLE	Telephone No. (703) 305-9795